École polytechnique fédérale de Lausanne

Master thesis, October 2023-March 2024

Mathematics section, Master in Computational Science and Engineering

Domain adaptation for birds monitoring from aerial surveys

Author:
Estelle Chabanel

Supervisors:
Benjamin Kellenberger,
Center for Biodiversity and Global Change, Yale University
Devis Tuia,
Environmental Computational Science and Earth Observation Laboratory, EPFL

Handover date:
February 29th, 2024

Abstract

Monitoring and census of birds are essential in ecology as birds populations provide an major indicator for the evaluation of ecosystem health. Recently, the development of aerial imagery coupled with advances in computer vision offered promising possibilities to ease birds monitoring at larger scales. In particular, deep neural networks that can learn to detect individual birds in images allowed the semi-automatic process of aerial datasets. However, most of the existing detection models for birds monitoring are trained on individual surveys, that focus on single species and ecosystems, which often makes them unsuitable for the process of new surveys. This inability of existing detectors to process new surveys highlights the domain shift problem caused by the diversity of birds habitats and species combined with the variety of surveys conditions. Training tailored detectors for new surveys is challenging since it requires a large quantity of annotated data, when the labeling process is laborious and costly. This work expose the challenges of training generalized models efficient across surveys, and study some adversarial domain adaptation methods to transfer knowledge from existing detection models to new aerial datasets. Using two pairs of datasets and the state-of-the-art YOLOv8 detector [78], we compare four domain adaptation architectures to address the domain shift problem encountered between the distinct birds surveys. We show the proposed architectures provide significant improvements on one of the domain shifts studied, doubling the average precision value when compared to the original YOLO detector. This proves the usability of domain adaptation methods in unsupervised setup to limit the labeling process costs for smaller domain shift in aerial birds surveys.

Résumé

L'études des populations d'oiseaux constitue un indicateur majeur pour l'évaluation de la santé des écosystèmes. Récemment, le développement de systèmes d'imagerie aérienne et les innovations en computer Vision ont grandement facilité ces études. De nombreux *neural networks* ont été développé et optimisé pour la détection d'oiseaux à partir d'images aériennes. Malheureusement, la majorité des systèmes de detection existants sont optimisés sur des sets d'images provenant d'uniques études, les rendant inneficaces lorsqu'appliqués sur de nouvelles études réalisées sur d'autres expèces ou écosystèmes avec différent matériels. De plus, entrainer de nouveaux modèles sur de nouvelles collection d'images nécessite une importante quantité de labels, rendant la tâche laborieuse et couteuse. Cette étude expose les défis soulevés par la création de modèles généralisés applicables sur différentes collection d'images et étudie les méthodes de *domain adaptation* pour optimiser des modèles existants sur de nouveaux datasets tout en limitant la quantité de labels nécessaire. En particulier, quatre méthodes sont dévelopées et comparées sur deux pairs de collections d'images aériennes provenant de différents système d'imageries, zones géographiques et présentant différentes espèces d'oiseaux. L'expérience réalisée démontre la possibilité d'utiliser les méthodes de *domain adaptation* sur certains datasets afin de limiter la quantité de labels nécessaire.

Contents

1 Introduction

The development of aerial imagery in recent years found numerous valuable applications in ecology and conservation studies. In particular, the usage of drones (Unmanned Aerial Systems and Vehicles (UAS, UAV), or Remotely Piloted Aircraft Systems (RPAS)), revolutionized data collection for both animals and lands studies, opening a all new range of possibilities for remote sensing [53], [82], [83]. In a context of constant declining biodiversity, these technologies enabled efficient study of species distributions and behaviors while addressing multiple challenges introduced by species monitoring [46], [32], [5]. Especially, airborne imagery offers low-costs observations at high resolutions while limiting human risks and impacts of intruders in observed species habitats. In the case of birds, the benefits of aerial imagery are particularly noteworthy as it allows high resolution surveys of inaccessible areas, when birds are sometimes difficult to detect by human eyes and can be found in a very wide range of environments [9]. Birds monitoring is a very important field of ecology, not only for endangered species conservation but also for ecosystems health assessment for which birds populations are a major indicator [16], [24].

Success of airborne imagery was supported by the development of tailored deep learning algorithms. Taking advantage of the new profusion of data, researchers are developing automatic systems for multiple tasks, including detection in remote sensing datasets [11], reducing analysis time, costs and difficulties. Deep learning object detection is a subfield of computer vision that aims to localize and classify objects within images. Coupled with aerial images, object detectors provide a very useful tool for animals monitoring. However, the majority of detectors developed for species monitoring are tailored for specific datasets, focusing on single species and geographical areas. As a result, these models are rarely suitable for inference on new acquired data [38], [9]. Developing new detectors for new datasets is challenging and comes at high costs. First, most deep learning models rely on the quality and quantity of annotated data. Labeling data is a laborious task and can be very expensive in terms of time and resources. Furthermore, developing efficient detectors for species monitoring requires technical resources and expertise, that are not always available for ecologists and biologists working on conservation. Birds monitoring is particularly impacted by these challenges. Considering the variety of bird species and living environments, detectors trained on a given aerial survey are very unlikely to perform on new data, recording distinct species and ecosystems. Moreover, the task of annotation can be extremely hazardous in images where birds are partly hidden by environmental elements.

One solution relies on the development of generalized models [37]. By learning general bird features shared across species, a generalized bird detector should be efficient on any kind of background and bird species. A general and resolution-invariant bird detector would then allow the quick and easy survey of new bird species or areas documented in new datasets, while limiting the costs of the labeling process and training new customized detectors.

Another solution is the development of an easy and effective transfer learning pipeline. This last one should take advantage of pre-existing labeled datasets and pre-trained bird detectors to optimize models on new unlabeled bird surveys. Domain adaptation for object detection [58], [79] explores such possibilities for datasets drawn from distinct probability distributions. This is especially relevant for bird detection from aerial images since birds surveys vary according to multiple characteristics (recorded species and ecosystems, weather conditions, imaging sensors, resolutions...) introducing a significant domain shift.

In this work, we explore the viability of both possibilities. Particularly, after assessing difficulties of the creation of a generalized model for bird detection, we focus on the potential of transfer learning via adversarial domain adaptation methods with domain classifiers. These last ones have already provided promising results on various domain shifts between datasets [26].

We construct our study following a review of relevant researches, presented in section 2. Present work relies on the use of multiple datasets gathered from previous works and recent state-of-the-art detection model YOLOv8 [78], [62]. Data and the detection model are presented in sections 3 and 4. Experimental approach and results are described in sections 5 and 6, and discussed in section 7.
The source code implemented for this study can be found in [8].

2 Literature review

2.1 Detection models

Detection systems have become an essential tool in biodiversity research and conservation works. Multiple detection architectures have been efficiently developed for animals detection and census, taking advantage of wildlife observation technologies like camera trap images [57], [69], [56], [3]. More recently, the arising popularity and accessibility of drones introduced a new source of imagery. Various detection models have been tailored for aerial images, focusing on various species like cattle [2], Savanna's animals [67], or other mammals [38], [51].

Many object detection frameworks rely on two-stages detection model architectures. Such models, like RCNN (Region-based Convolutional Neural Network, [20]) Fast-RCNN [19] or Faster-RCNN [65] divide the detection problem in two tasks, executed one after the other. In the first step, models generate region proposals. To do so, RCNN and Fast-RCNN use a selective search algorithm [77] while Faster-RCNN and more recent object detectors rely on a deep learning model like the Region Proposal Network (RPN). This last one is a fully-convolutional neural network that takes an image convolutional feature maps as input and outputs rectangular region proposals, using a sliding window process. For each sliding window position, reference boxes of different scales and aspect ratios (called anchors) are used to generate region proposals and corresponding probability of the proposal to contain an object, the objectness score. In the second stage, a Convolutional Neural Network (CNN) performs classification and bounding box regression of the region proposals. For each proposal, the CNN retrieve corresponding feature maps to adjust anchor boxes and objectness scores, and classify objects.

On the other hand, one stage detection systems like SSD [47], RetinaNet [44] and YOLO [62] address the task of detection using a single deep neural network, skipping the region proposal stage to directly localize and classify object from one pass of the input image. In particular, the YOLO models, standing for "You Only Look Once" are frequently updated. The most recent version, YOLOv8 [78], predicts accurate detections on the Pascal VOC dataset [14]. The model consists of a CNN that predicts bounding boxes and their class probabilities at once. To do so, the architecture is divided into a backbone that extracts features from input images and a Head that uses these features to predict bounding boxes, their confidence scores and class probabilities simultaneously. This architecture allows faster predictions than two-stage detectors.

2.2 Bird detection from aerial images

Aerial imagery is particularly beneficial for birds surveys, as it greatly eases the data collection process and expand surveys to inaccessible areas. Hence, multiple detection architectures have been tailored to the task of bird detections from aerial images, tackling various challenges specific to the task [9], [48], [31].

Hong et al. [31] compared the performances of multiple existing CNN-based object detectors on a dataset composed of both wild and decoy birds in UAV images of lakes and farmlands in South Korea. The results demonstrate existing architectures capacity to learn birds features from UAV aerial imagery. Especially, both two-stages and one-stage detectors were tested, all presenting accurate detections, the YOLO 9000 [63] and YOLOv3 [64] performing generally faster. Kabra et al. [34] focused on waterbirds and developed an object detection pipeline using two existing detectors, RetinaNet [44] and Faster-RCNN [65]. The proposed pipeline revealed potential to learn distinct species features by efficiently distinguishing between at least two species of waterbirds of the Texas coast. Albatross and penguins of the Falkland islands colonies were surveyed by Hayes et al. [27], using the RetinaNet architecture and providing high accuracy census results. The final architecture overcomes challenges of bird census in colonies, specifically the difficulty to distinguish between distinct species and identify units in huge flocks of birds. Furthermore, the issue of low resolution images has been tackled by Li et al. [41], through a low to high resolution images mapping. A Single Image Super-Resolution (SISR) deep neural network learns a low to high resolution mapping and reconstruct high resolution images, that are then used for object detection. Both two-stages and one-stage detectors provide higher detection precision on reconstructed high resolution images than on input low resolution images.

These few works are just a small portion of existing research and analysis on the task of bird detection from aerial images. They highlight some of the challenges of the task and demonstrate detection models capacity to overcome them. However, most of the architectures developed focus on a single survey, presenting images taken from the same imaging sensor, and including a small number of species only and a single ecosystem. In inference mode, those models fail to provide accurate detection when applied on new surveys realized in distinct conditions and focusing on other ecosystems or species. Weinstein et al. [81] tried to address this issue

by training a generalized model using datasets from various studies on multiple ecosystems. The resulting set of images accounted for various geographical areas, scenes, species and image resolutions, leading the model to detect birds of multiple species in multiple habitats. Following a cross-validation process, generalized models are trained on a combination of multiple datasets and evaluated on a distinct one. The resulting models present improved F1-score on most of the testing datasets, when compared to the detection model trained on very few annotations of the testing dataset only. However, for some models, performances remain relatively low and only the introduction of a limited amount of annotations in a semi-supervised setup allows the models to reach higher F1 scores. Moreover, performances are very uneven across tested datasets, revealing difficulties to generalized across all habitats.

2.3 Domain Shift problem

The domain shift problem [40], [15] arises when the training data and the testing data of a machine learning model are drawn from different distributions, significantly impacting the model performances on the testing set. In the case of bird detection, this is often the case as images can present various ecosystems, geographical areas and species, or come from different imaging sensors. The cause and types of variation in images distributions are then numerous: variation in scenes, meteorological conditions, resolutions and altitudes, cameras sensors, settings and quality... Annotations distributions can also differ when distinct species are present in training and test sets or when species present imbalances between both sets.
Multiple methods have been designed to address this domain shift problem between the two domains: the source domain being the one encountered in the training set while the target domain refers to the testing set on which performances need to be improved. Specifically, domain adaptation methods have been extensively studied and proven efficient for the tasks of classification and segmentation [15]. The object detection task, however, involves new challenges and has aroused interest in recent years. Referred as DAOD, domain adaptation techniques for object detection have been surveyed in [58] and [79].
We present here a non-exhaustive description of possible methods to address the domain shift problem encountered in the bird detection task in aerial images.

2.3.1 Continual Learning

Continual learning [1], also referred as incremental learning and life-long learning consists in teaching a model a number of tasks (detecting birds in distinct domains or datasets) sequentially without forgetting the knowledge learnt in preceding tasks. Although not originally designed for domain adaptation, continual learning has been used to address the domain shift issue like in Prasanna et al. [60]. The architecture relies on weights-based pruning techniques to free up a portion of parameters of the network. These freed up parameters are then iteratively updapted to learn a new task (detection on a new domain, the target domain), keeping the other parameters fixed and taking advantage of previously learnt features. A big advantage of the pruning-based methods is the non-necessity to store original data, the tasks being learnt sequentially, keeping track of the weights assignments. However, a drawback is the need to store parameters and tasks/domains correspondence, and to apply filters on parameters when using the model for predictions. Moreover, the pruning of parameters is followed by a drop in performances. The usability of this method is then limited by the drop in performances the model can endure on the source domain. Finally, such Continual learning methods were mainly derived for classification task while the application of continual learning to object detection has been less studied.

2.3.2 Feature alignment and adversarial domain adaptation

A more popular range of methods for object detection approach domain adaptation through domain invariant feature learning. In CNN-based object detection systems, features are extracted from input images and encode all types of information relevant to the object to detect. The features are then used by the models to localize and classify objects. In DAOD feature alignment methods, features of both the source and target domains are extracted from a given layer of the neural network and are aligned using a divergence criterion. Multiple divergence criteria have been tested such as the Coral Loss [73] that aligns the covariance of the source and target non-spatial features output by a classifier network. Other popular divergence criteria are traditional distance metrics between source and target features probability distributions, like the Maximum Mean Discrepancy [49], [70] and the Wasserstein metric [71]; or the Contrastive Domain Discrepancy [36] that accounts for class information and minimizes the difference between intra-class distributions across domains while maximizing the difference of inter-class distributions. One important benefit of such methods is that if not all originally designed for detection, they can easily be adapted to various architectures as soon as the model can be divided

into a feature extractor and a detection head.

Another possibility for domain invariant feature learning is adversarial domain adaptation [26] which constitutes one of the most common approach in DAOD. Again, the model is trained to efficiently complete the main task (bird detection) on source examples based on the extraction of domain invariant features. However, the model learns domain invariant features through the help of a domain classifier that takes features as input and returns the predicted domain, source or target. During training, the domain classifier is led to provide less and less accurate domain predictions, while the detection losses of the object detector are minimized to obtain an efficient model on both source and target domain. This leads the features encoder to provide more and more indistinguishable features between source and target domains. The adversarial domain adaptation architecture can be trained in an unsupervised manner, without target labels, only knowledge about provenance domain of the examples being necessary. Moreover, this method has already shown promising results for both classification and detection [17], [58].

Wei et al. [80] confirmed the usability of the YOLO model for cross-domain object detection tasks by proposing an extended architecture YOLO-G, based on YOLOv5. The main addition of the new model is an unsupervised adversarial branch that performs feature alignment from the output of the backbone. It is composed of a domain classifier preceded by a gradient reversal layer [17]. This last one is responsible for the double objective of minimization-maximization of the classifier loss. The domain classifier loss is minimized in the local backpropagation of the adversarial branch and maximized in the backward pass towards the feature extractor network. The overall model is trained using the weighted sum of the binary cross-entropy loss of the domain classifier and the original detection loss of the YOLOv5 model. Hnewa et al. [29] implemented a similar method. Taking advantage of the three scaled features extraction of the backbone, the YOLOv4 model is extended with a Domain Adaptive Network composed of several adversarial branches to predict the examples' domains from each of the three scaled feature outputs of the backbone or feature extractor. The model then learns domain invariant features at three different scales, features that are then fed to the detection head.

In DA-YOLO designed by Zhang et al. [85], YOLOv3 is modified to learn both domain invariant image features and domain invariant instance features. To do so, a first Regressive Image Alignment block applies three image-level domain classifiers to the intermediate image-level features outputed by the backbone, as in Hnewa et al. [29]. Then, a Multi-scale Instance Alignment block takes the detection results of the three scaled detection layers to extract instance level features and apply instance level domain classifiers. The introduction of this second alignment step was introduced by Chen et al. [10] for the Faster RCNN architecture [66], taking advantage of the two-stages characteristic of the detector. However, the efficient method is here adapted to the one-stage detection model, using ROI pooling to map detection bounding boxes to portion of the feature maps. The training loss of the overall model is then the sum of both image and instance domain classifiers losses, and the YOLOv3 detection loss.

2.3.3 Student-teacher framework

A second very popular approach for DAOD relies on the adaptation of the student-teacher method originally designed for semi-supervised learning. The small "student" model learns to mimic the larger "teacher" model using a distillation algorithm [23], [28]. By doing so, the student model learns to generalize in the same way as the teacher model, allowing it to often generalize and perform better on unknown data than the original model. Some works took advantage of this last characteristic to adapt knowledge distillation techniques to incremental learning [72] or domain adaptation [25], [76].

Most recent approaches to domain adaptation get inspired from the Mean Teacher architecture [75], where the student and teacher model are built with an identical architecture and learn to provide consistent predictions for slightly different inputs. Cai et al. [7] proposed Mean Teacher with Object Relations (MTOR) that trains the teacher on unlabeled target data to generate pseudo-labels, while the student is trained on both unlabeled target samples and labeled source samples, adjusting the student-teacher framework for unsupervised domain adaptation. Other works extended the Mean teacher network to unsupervised learning by creating fake source and target samples, like the Unbiased Mean Teacher (UMT) [13] that relies on CycleGAN or SSDA-YOLO [88] that generates fake examples using image translator [59].

Finally, some works tried to combine advantages of both adversarial DAOD and the student-teacher framework, by incorporating a domain classifier in the student or teacher model, presenting improved performances on the tested target dataset when compared to pre-existing DAOD like UMT [43], [84], [74].

3 Data

In order to assess challenges of generalized models for bird detection and study domain adaptation methods, data used for training must reflect the diversity of aerial imagery surveys. Hence, data were gathered from multiple sources, corresponding to different drones and cameras, resolutions, geographical areas, species and sceneries, the only required common ground being the aerial images type. Images were taken from previous works, online available datasets, and from the large collection of aerial images for wildlife studies gathered in [5].

Table 1 shows characteristics of all the datasets gathered for this study. Datasets are named after the name of the author, or the name given in the original studies. A total of 8 sources have been collected, resulting in 2 672 airbone images and 64 525 birds annotations. Images are encoded in the RGB format, with pixel values in the range [0, 255] except for the Waterfowls [3] dataset composed of thermal images encoded in grayscale, with pixel values in the same range, [0, 255].

The final set of images covers a large range of geographical areas and landscapes, from river banks in Poland to forest canopy images of the Palmyra Atoll islands. Distinct species are also represented in the final set of data even though most surveys focus on waterfowls and penguins images.
To give a visual overview of the diversity of images gathered, Figure 2 shows two example images for each source. Images are presented after preprocessing, along with their annotations in green bounding boxes.

Source	Geographical area	Bird species	# images	# annotations	Ground sampling distance	Image size in pixels
Pfeifer [6]	South Shetland islands, Antarctica	penguin	846	27 226	1-3.4 cm	450×450
Penguins [1]	Islands off the Antarctic Peninsula	penguin	235	12 892	1-2 cm	500×500
Palmyra [1]	Palmyra Atoll National Wildlife Refuge, Northern Line Island Archipelago, Pacific Ocean	seabird	452	3 287	0.7 cm	1400×1400
McKellar [1]	Saskatchewan, Canada	grebe, Franklin's gull, black tern, Forster's tern, black-crowned night-heron	404	3 327	6–16 cm and 0.8–2 cm	900×900
Poland [1]	Nothern Poland	black-headed gull, common Tern, gray heron, mallard, mute swan, tufted duck, greater scaup, Eurasian wigeon, Eurasian teal, northern pintail	381	8 817	unknown	1200×1200
Hayes [2]	Jason and Grand Jason Islands, Falkland Islands	albatross, penguin	3 947	44 966	unknown	1000×1000
Terns [4]	Mauritania and Guinea	royal tern, caspian tern, slender-billed gull, gray headed gull, great cormorant, great white pelican, others	1	21 516	~ 1cm	27569 ×28814
Waterfowls [3]	Wetland area of Nebraska, US	waterfowls	354	8 976	7.5 cm	512×640

Table 1: Characteristics of the different sources of aerial images used throughout the study.

3.1 Data preprocessing

A majority of the datasets gathered were published by researchers following their work on detection models. As a result, most of the datasets were already partly preprocessed and came as a set of squared images of given sizes. However, to homogenize the final dataset and make it compatible with the YOLO model, a second preprocessing stage is applied on all the datasets.

Bird annotations are homogenized to the YOLO labels format. Each annotation is converted to a rectangle bounding box defined by its normalized top left corner coordinates, height and width. For datasets with only two coordinates-point annotations defined, the bounding boxes width and height is set approximately by observations on a subset of images.

Images are cropped following a sliding window process, resulting in patches of size multiple of 32, which meets the need of the YOLO models. In order to preserve the original aspect ratio of the birds in the images and diversity of our final dataset, the patch size is defined as the closest smaller multiple of 32 to the original image size. At the edges of the new patches, birds for which less than 70% of the bounding box appears in the new patch are filtered out. Moreover, to eliminate existing edge cases observed in the annotations, for each distinct dataset, annotations smaller than half of the mean bounding box size are also deleted. This process resulted in a total of 18 223 patches and 157 090 bird annotations.

Many sources include a lot of background patches (that sometimes account for more than 75% of the dataset images, like in the Palmyra and Terns datasets). To prevent possible impact on the models performances, the amount of background patches is limited to 10% of the total amount of patches, within each independent dataset. This 10% ratio is within the recommended range of background images for detection task [78], and appeared to be an efficient ratio for our model after multiple experiments. The resulting ensemble includes 11184 patches with 157090 bird annotations and will be referred as the "global dataset" in the following of the study.

Finally, within each dataset, images are split into train, validation and test sets. To avoid any overlap across the three sets and bias in the models evaluation, train, validation and test sets are constructed in a way to cover distinct flight times and geographical areas when possible. For a few sources, information on the conditions of the surveys are missing and the split follows a basic random split procedure. For all sources, the final split approximately corresponds to a 70/10/20% split.

3.2 Datasets statistics

To become familiar with the data and assess the variety of the aerial images gathered, some statistics and important figures are gathered in Table 2.

The distribution of birds in patches is also studied across distinct sources and splits. The distributions of the global dataset formed by the totality of the sources gathered is shown per split in Figure 1. Histograms of birds distribution for some sources studied in details in the following of the experiment are available in Appendix A. These distributions, along with the mean number of birds per patch given in Table 2 illustrate the diversity presented by the surveys. The mean number of birds per patch is included in a very large interval (from 1.83 for the Palmyra dataset to 48.31 in the Terns dataset). The mean pixel size of bird annotations demonstrate both the variety of species in the global dataset and the large range of resolutions of the ensemble of images. This variety of bird scales present an important challenge for the training of detection models and will be discussed further in the following of the experiment.

Drone imagery systems allowing the survey of large areas, independent datasets also present a large variety in terms of backgrounds, colors and sometimes species. This intra-datasets variety is particularly visible for the Palmyra and Poland datasets in the two example images provided in Figures 2b and 2e. The mean and standard deviation of pixel values across channels also demonstrate the inter and intra-variety of the datasets.

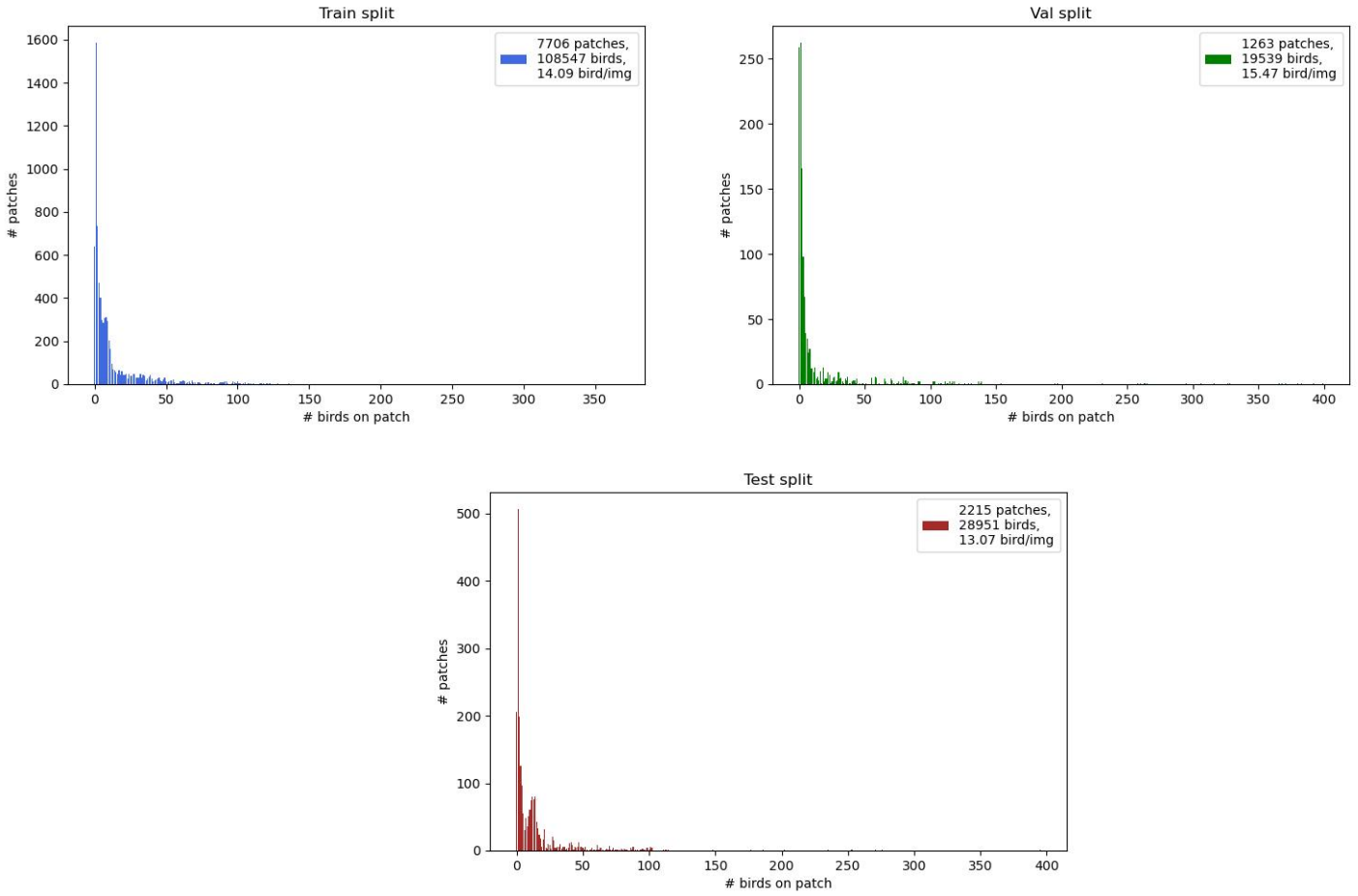
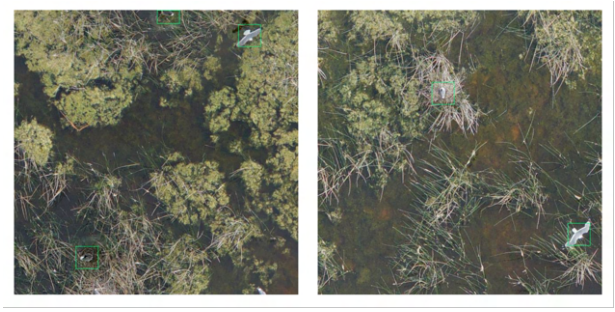


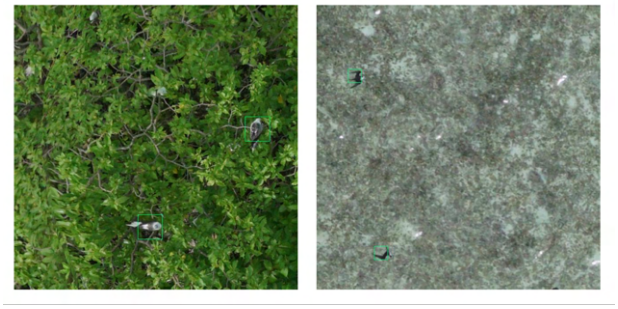
Figure 1: Bird distributions across patches of the three sets (train, val, test) of the global dataset.

Source	Patch width in pixel	# birds	# patches	Mean #bird/patch	# bckgd patch (%)	Mean bird size in pixel	Mean pixel values	Std of pixel values
McKellar	640	2329	974	2.39	98 (10.1%)	48×48	[107.3, 106.9, 97.3]	[30.7, 29.5, 25.9]
Palmyra	640	3194	1746	1.83	175 (10.0%)	37×37	[89.3, 108.4, 82.9]	30.1, 31.5, 28.4
Penguins	460	12003	940	12.77	78 (8.3%)	15×15	[134.0, 132.7, 133.0]	[32.1, 32.1, 32.6]
Pfeifer	448	84902	3344	25.39	335 (10.0%)	25×25	[134.0, 132.7, 133.0]	[32.1, 32.1, 32.6]
Poland	640	4811	946	5.09	95 (10.0%)	59×59	[83.6, 85.9, 82.8]	[29.8, 29.6, 30.1]
Hayes albatross	640	13887	2018	6.88	202 (10.0%)	38×38	[121.2, 118.2, 112.3]	[56.4, 56.1, 55.2]
Terns	640	21351	442	48.31	45 (10.2%)	40×40	[153.1, 118.6, 76.2]	[26.0, 26.1, 25.9]
Waterfowls	512	14613	774	18.88	78 (10.1%)	7×7	106.2	27.4

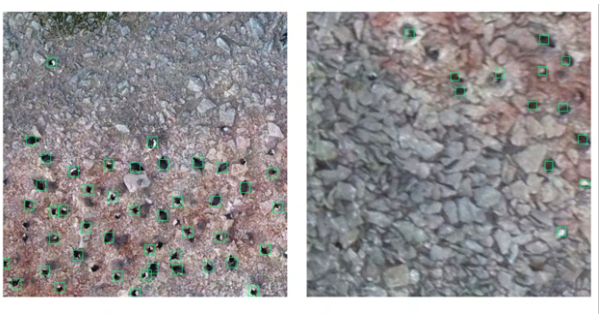
Table 2: Characteristics and statistics of the distinct datasets of aerial images after pre-processing.



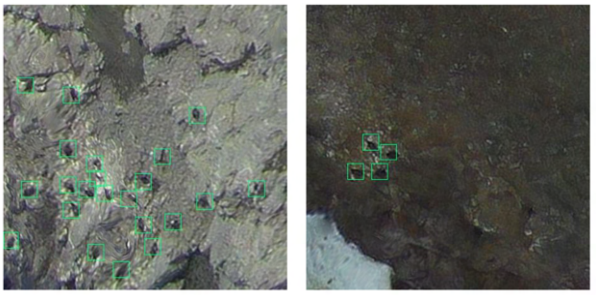
(a) McKellar



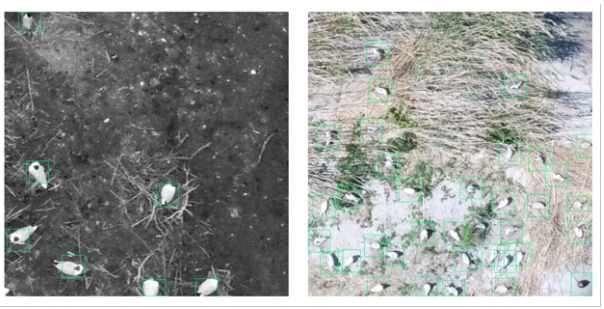
(b) Palmyra



(c) Penguins

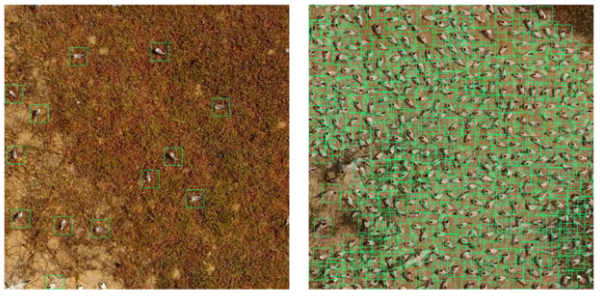


(d) Pfeifer

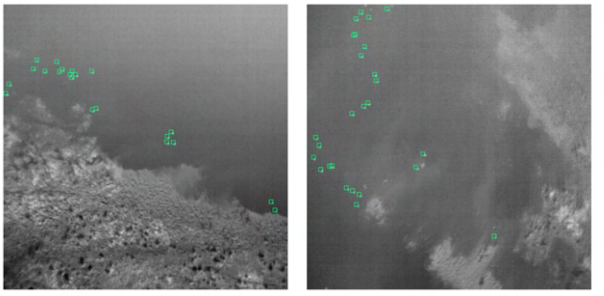


(e) Poland

(f) Hayes albatross



(g) Terns



(h) Waterfowls

Figure 2: Example of patches from the different sources after preprocessing.

4 YOLO model

All models and experiments in this study are derived from the last version of the YOLO models [62], YOLOv8 [78] designed for object detection. Its architecture is briefly described below.

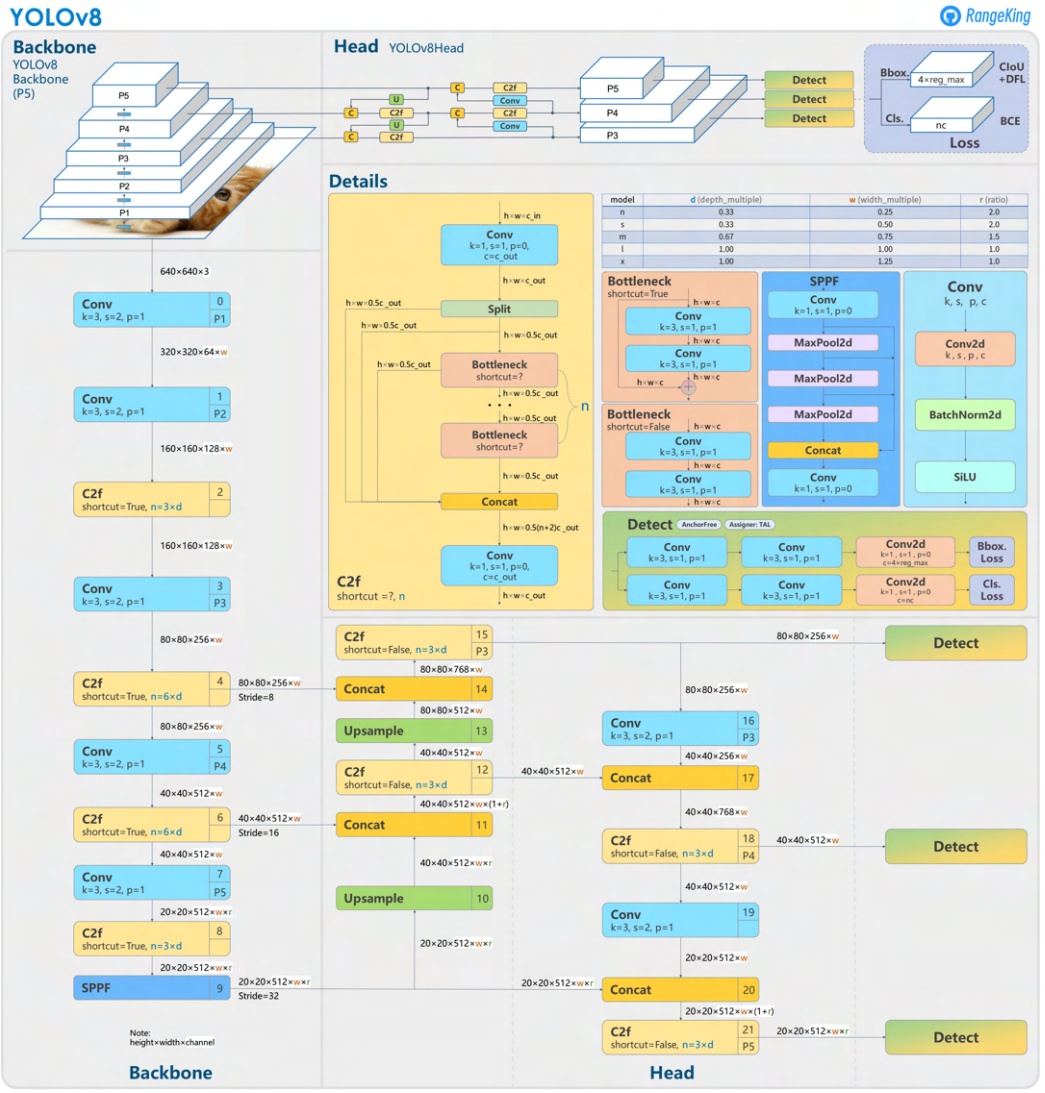


Figure 3: Scheme of the YOLOv8 architecture for object detection [61].

4.1 YOLOv8 architecture

The overall architecture of the model is presented in Figure 3. The YOLOv8 model can be divided in three main parts: the backbone, the neck and the head. The backbone is a modified version of the CSPDarknet53 architecture, a convolutional neural network for object detection introduced in YOLOv4 [4]. It takes the images as input and outputs three feature maps, at three different scales. The Neck, sometimes incorporated in the Head, combine information of the feature maps before feeding them to the detection Head that performs bounding box predictions and classification of objects.

As mentioned before, the particularity of the YOLO models is in the simultaneous predictions of bounding boxes and classification of objects. To do so, the input image is divided into a $S \times S$ grid of cells. The model predicts B bounding boxes and their objectness scores, and C class probabilities (corresponding to the total number of class C), for each cell. These predictions are combined in the training loss used to train the model. The computation of the loss is described in detail below. In inference mode, the bounding boxes objectness scores and the class probabilities are multiplied to obtain class-specific confidence scores for each bounding boxes. This last one represents both the probability of the bounding box to contain an object of the given class and the accuracy of the predicted box.

4.2 Training losses

YOLOv8 models are trained using a weighted sum of the bounding box loss and the classification loss. The bounding box loss is computed via Complete Intersection Over Union [86], [87] and Distribution Focal Loss [42], while the binary cross entropy loss accounts for the classification loss.

Complete Intersection over Union

The Complete Intersection over Union loss (CIoU) was introduced to account for three geometric factors in bounding box regression. The first one, the overlap area, is computed using the traditional Intersection over Union loss (IoU), $S(A, B)$ that returns 0 for a perfect match between ground truth A and predicted bounding box B and 1 for no overlap. The second factor, Normalized Central Point Distance, $D(A, B)$ accounts for the normalized distance between the predicted bounding box center and the ground truth center. It leads to better bounding boxes predictions by getting their centers closer to the ground truth centers when minimized. Finally, the CIoU loss introduces the aspect ratio $V(A, B)$ that impose consistency of predicted and ground truth bounding boxes aspect ratio. Final computation of the CIoU loss is given in equation (1):

$$\text{CIoU}(A, B) = S(A, B) + D(A, B) + V(A, B) \quad \text{with} \quad \begin{cases} S(A, B) & = 1 - \text{IoU}(A, B) = \frac{|A \bigcap B|}{|A \bigcup B|} \\ D(A, B) & = \frac{\rho^2(a,b)}{c^2} \\ V(A, B) & = \alpha \frac{4}{\pi^2} \left(\arctan\left(\frac{w^A}{h^A}\right) - \arctan\left(\frac{w^B}{h^B}\right) \right) \end{cases} \tag{1}$$

with A, B the ground truth and predicted bounding boxes, and a, b their respective center; $\rho(\cdot)$ denotes the Euclidian distance and c is the diagonal length of the smallest box enclosing both A and B bounding boxes. In the aspect ratio computation, α is a trade-off parameter used to give less importance to the geometric factor for low IoU values.

Distribution Focal Loss

In bounding box regression, the Distribution Focal loss is applied in addition to the CIoU loss to address uncertainties in predicted boxes locations. To do so, it learns continuous bounding boxes distributions instead of fixed bounding boxes.

Binary Cross-Entropy Loss

For classification, the traditional binary cross entropy loss is used regardless of the number of classes C. This means the detection Head can be seen as a combination of C independent classifiers. For each predicted bounding box, the probability to belong to each one of the classes is computed separately. Hence, multiple labels can be assigned to the same box in case of overlapping objects. The binary cross entropy loss is computed on the sigmoid of the logits output of the detection Head. The final computation is given in equation (2)

$$\text{BCE}(x, t) = \sum_{n=1}^{N} t_n \log \sigma(x_n) + (1 - t_n) \log(1 - \sigma(x_n)) \tag{2}$$

where x_n is the the logit output for example n, with classification ground truth t_n and $\sigma(\cdot)$ denotes the sigmoid function.

The three losses are summed up to give the final loss minimized in the backward loop. The final loss is given in equation (3) with λ, β, γ the corresponding losses weights. YOLOv8's default weights are set to $\lambda = 7.5, \beta = 1.5$ and $\gamma = 0.5$.

$$\mathcal{L} = \lambda * \text{CIoU} + \beta * \text{DFL} + \gamma * \text{BCE} \tag{3}$$

4.3 Pre-trained models

Finally, Ultralytics [78] proposes YOLOv8 pretrained models. Those models are trained on two distinct datasets, the 2017 version of MS COCO and OpenImages v7. The MS COCO dataset [45] contains 328 000 photos with 2.5 million labeled instances of 80 distinct categories while OpenImages v7 [22] gathers 1.9 million images with 16 millions bounding boxes for 600 object classes. MS COCO proposes more general categories, gathered in 11 sub-categories, visible on Figure 4. In particular, bird is one of the category annotated in the MS COCO dataset. On the other hand, the OpenImages v7 dataset contains a general category "bird" but also more precise categories and distinct species of birds like "falcon", "goose", "swan" or "turkey". Hence, for our task of single-bird class detection, pre-trained models on the MS COCO dataset seems like a more natural choice since the models could have already learnt general bird features.

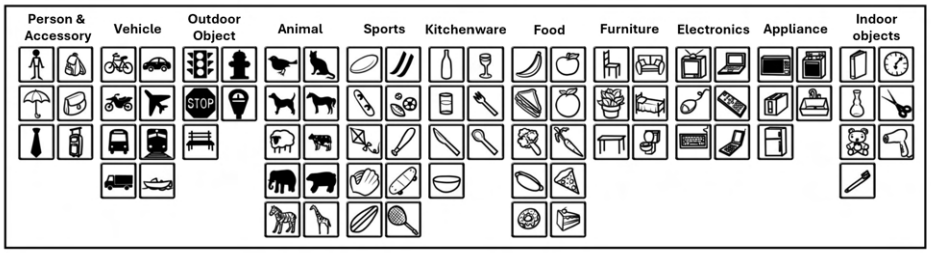


Figure 4: Icons of the 80 categories annotated in the 2017 MS COCO dataset, grouped in 11 super-categories [45].

Moreover, the YOLOv8 models come in five distinct sizes, nano, small, medium, large and extra large. Depending on the size of the dataset and the difficulty of the task to learn, bigger networks can help the model learn more efficient features. In order to compare results across different datasets combination, the YOLOv8m model, corresponding to the medium size will be used throughout this experiment. It consists of 295 layers and 25 902 640 parameters.

Performances of the YOLOv8 models trained on the MS COCO dataset are visible in Figure 5.

Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Figure 5: Performances of the pre-trained YOLOv8 detection models, evaluated on the MS COCO dataset [78].

5 Methods

5.1 Baseline models and training process

In order to highlight the challenges of creating a generalized bird detector, a first global model is trained on the global dataset training set (composed of the training sets of all the sources gathered and presented in section 3). The detection model used is the original YOLOv8 architecture presented in section 4 and initialized with the YOLOv8m pre-trained weights obtained after training on the MS COCO dataset [45].

Then, in order to get a better understanding of models behaviors, multiple models are trained on subsets of sources. Particularly, we identify datasets combinations that highlight the domain shift problem, showing important disparities in performances across independent sources. Those subsets establish a ground for exploration of domain adaptation methods.

We describe below the training process used to train the global model. The same process is used for the smaller models, with some changes in parameters. All training parameters are given in Appendix B.1.

5.1.1 Training process

Training is performed on an NVIDIA GeForce RTX GPU with 32 GB of memory. The model is trained for 200 epochs, with a patience of 50 epochs, using the SGD optimizer and 0.01 initial learning rate. During training, the scheduler linearly decreases this learning rate until it reaches 0.01 times the initial learning rate at the end of the last epoch.

A probability-based augmentation process is applied during training. For each batch of 32 training examples, some images are replaced by a transformed version, according to predefined probabilities. The transformation applied are chosen according to the specificities of our datasets and aimed task. In order to get a robust model and augment generality of the final datasets, images can be rotated, flipped horizontally and vertically and rescaled to simulate various scales of annotations within individual datasets. Images hue, saturation and lightness can also be modified according to pre-defined fractions. Finally, patches smaller than 640x640 pixels are resized to 640 pixels width and height.
The model is evaluated on the validation set at the end of each epoch to further tune training parameters. At the end of training, weights corresponding to the best epoch are saved as the final model. The best epoch is the one reaching the best fitness on validation set, with fitness computed as the weighted sum of two metrics, mAP@0.5 and mAP@0.5:0.95. mAP@0.5 corresponds to the mean average precision when the matching IoU threshold is set to 0.5, while mAP@0.5:0.95 represents the mean of mean average precision values for different IoU thresholds from 0.5 to 0.95. Their respective weights for the fitness computation are set to to 0.1 and 0.9 respectively.

Training parameters are defined after a grid search using YOLOv8's evaluation criterion: for multiple combinations of values, the parameter set corresponding to the model with highest fitness metric is selected.

These same training and parameters tuning processes are used throughout the following of the experiments.

5.2 Features alignment using l^2-norm

Following the researches presented in section 2, multiple domain adaptation methods are studied and applied to the YOLOv8 model.

The first domain adaptation method tested is features alignment using a distance metric between source and target domains. Similarly to the Coral loss [73] and Maximum Mean Discrepancy [49] ideas, source and target features are aligned using the l^2-norm. To do so, an adaptive average pooling layer is applied to the output of the YOLOv8's backbone. This layer allows to get rid of the spatial dimension of the features and compute the l^2-norm between each source and target examples of the batch. The mean of the norm values is then added in the final weighted sum of the model's losses, with associated weight α. The overall loss is given in equation (4). Its minimization in the backward loop leads the model towards learning similar features for source and target domains.

$$\mathcal{L} = \lambda \text{CIoU} + \beta \text{DFL} + \gamma \text{BCE} + \alpha \frac{1}{\text{NM}} \sum_{\text{N, M}} \|\mathbf{y_n} - \mathbf{z_m}\|_2 \tag{4}$$

where N and M are the total number of source and target samples respectively, y_n and z_m are the last layer output of the bakbone of the n-th and m-th source and target sample. $\|\cdot\|$ denotes the l^2-norm.

This method constitutes a proof for features alignment possibilities and will serve as a baseline for comparison of domain adaptation methods. The method is referred as the L2-norm minimization in the following of the experiment.

5.3 Adversarial domain adaptation

Several architectures based on adversarial domain adaptation using domain classifiers are compared in this study.

5.3.1 Single domain classifier

First, the YOLOv8m model is modified to embed a single domain classifier. The architecture is presented in Figure 6, all output sizes are given in the same format, $C \times H \times W$.

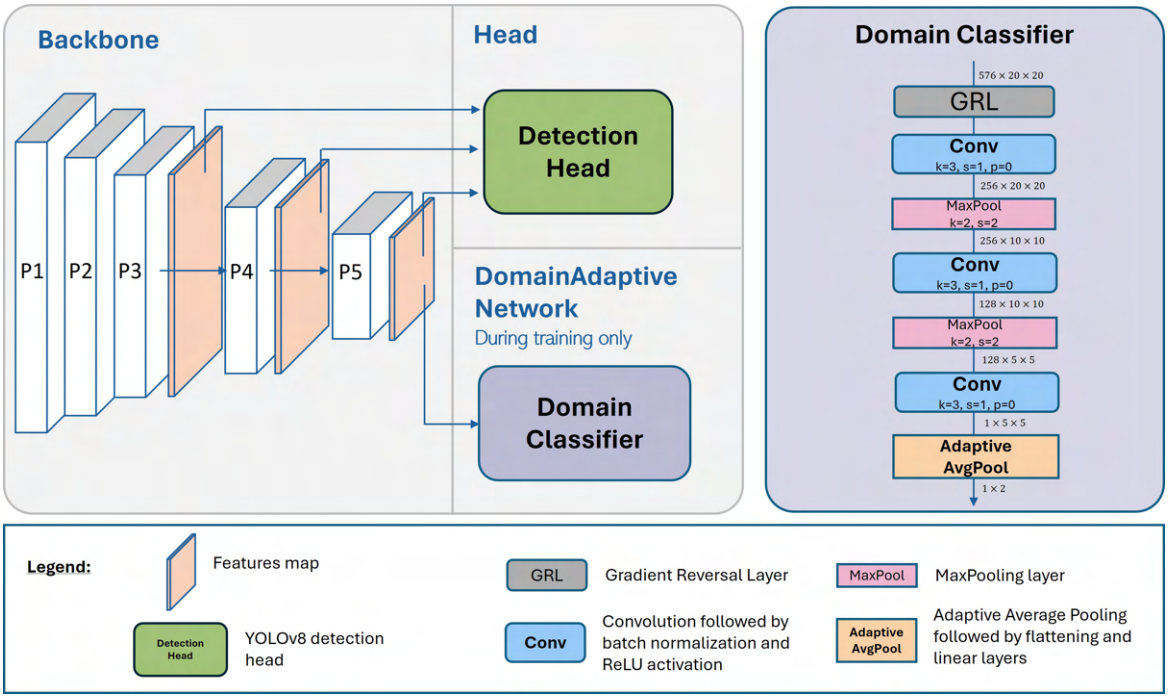


Figure 6: Architecture of the proposed single Domain Classifier architecture. Left scheme gives an overview of the model's architecture, with original YOLOv8's backbone and head, and the Domain Adaptive network extension. The right scheme shows the detailed structure of the Domain Classifier or Domain Adaptive network.

The backbone encoder of the original YOLOv8m model outputs a 576×20×20 feature map, that is then fed to both the head network of the original YOLOv8 model and a domain classifier subnetwork returning domain class probabilities. This domain classifier consists of a gradient reversal layer [17], three convolutional blocks with batch normalization and ReLU activation functions, and an adaptive average pooling layer followed by a flatten and a dense layer. It outputs classes logits probabilities of source and target domains for each training example of the current batch. The domain classifier loss is the cross-entropy loss computed on the softmax activation of these logits. The computation is given in equation (5):

$$\mathcal{L}_{DC} = \sum_{n=1}^{N} -t_n \log \frac{\exp(x_n)}{\sum_{c=1}^{C} \exp(x_{n,c})} \tag{5}$$

with x_n the logit output of example n, with classification target t_n, and $C = 2$ the number of classes.
By changing the sign of this loss function in the backward pass, the gradient reversal layer [17] allows to minimize the loss locally in the domain adaptative subnetwork and maximize it in the pass towards the backbone.

This dual-objective optimization leads the backbone to learn domain invariant features.

The final model is composed of 309 layers, with 27 481 226 parameters to optimize. The overall model is then trained for 200 epochs with a patience of 30 epochs, using a weighted sum of the domain classifier loss $\mathcal{L}_{DC}$ and the original losses of the YOLOv8 model, described in section 4.2. Final loss is given in equation (6).

$$\mathcal{L} = \lambda \text{CIoU} + \beta \text{DFL} + \gamma \text{BCE} + \alpha \mathcal{L}_{DC} \tag{6}$$

with α the domain classifier loss weight. This last one constitutes a new hyper parameter to determine by grid search.

In inference mode, layers corresponding to the Domain Classifier architecture are frozen, so that predictions are made from the original YOLOv8 model which learnt to provide domain invariant features.

5.3.2 Multiple domain classifiers

In order to take advantage of the YOLOv8 architecture and improve the adaptation performances, the previous model is extended with two new domain classifiers. These last ones are applied on the intermediate feature maps of the backbone of sizes $192 \times 80 \times 80$ and $384 \times 40 \times 40$. Their architecture is similar to the first domain classifier detailed above, only the kernel and output sizes of the convolutional layers are changed. The modified architecture is shown in Figure 7.

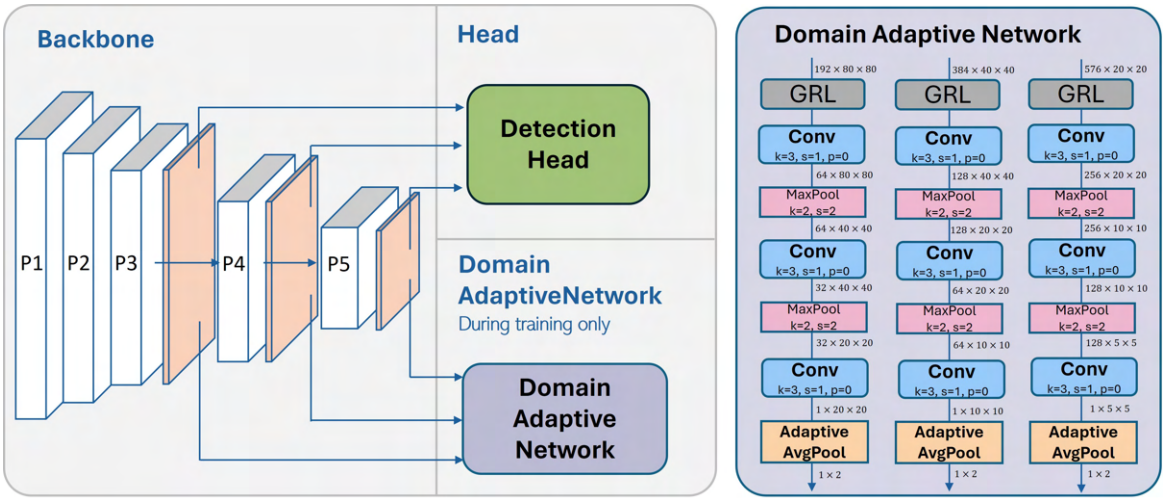


Figure 7: Architecture of the proposed Multi-Domain Classifiers architecture. Left scheme gives an overview of the model's architecture, with original YOLOv8's backbone and Head, and the Domain Adaptive network extension. The right scheme shows the detailed structure of the three domain classifier components of the Domain Adaptive network.

The resulting model is made of 337 layers, with 28 128 088 parameters. The training process is the same as for the single domain classifier. Only the final loss is updated. Each domain classifier giving a distinct classification loss $\mathcal{L}_{DC_i}$, three additional losses are added in the weighted final loss (equation (7)). Moreover, a distinct weight α_i is assigned to each one of these new losses. The three domain classifier losses $\mathcal{L}_{DC_i}$ are computed using the Binary Cross entropy loss as given in equation (5).

$$\mathcal{L} = \mathcal{L}_{YOLO} + \alpha_1 \mathcal{L}_{DC_1} + \alpha_2 \mathcal{L}_{DC_2} + \alpha_3 \mathcal{L}_{DC_3} \tag{7}$$

This new architecture takes advantage of the YOLO architecture and helps solving possible vanishing gradient problems. It increases the impact of the domain adaptation subnetwork on the smaller feature scales and leads the model to learn domain invariant features in earlier layers of the backbone.

5.3.3 Further studies on the domain classifier architecture

Extending the idea of the multiple scales domain adaptive network, further architectures are tested. A unique domain classifier is implemented, taking the three scaled feature maps of the backbone as inputs. This method allows to unify domain predictions among the different scales. After several tests for the architecture of the domain adaptive network, the most promising one is presented in Figure 8. The architecture of the overall model with backbone and detection Head is the same as in Figure 7, only the architecture of the domain adaptive subnetwork is modified. As in the previous architecture, a gradient reversal layer is applied on each feature map, followed by several convolutions and max pooling layers. The distinct features are then concatenated at different scales. The final concatenation is followed by several convolutions to further reduce dimensions and a final adaptive average pooling layer outputs domain classes logits.

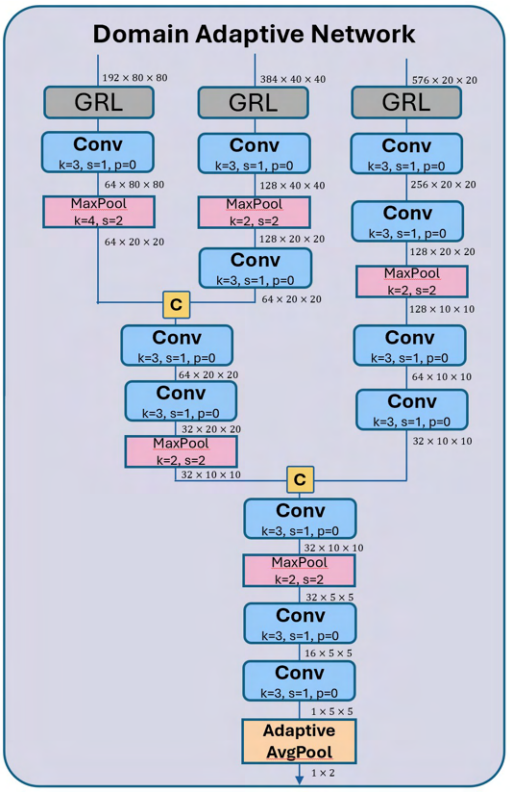


Figure 8: Detailed architecture of the proposed Domain Adaptive Network for the Multi-Features Domain Classifier. The network takes the three scaled feature maps of YOLOv8's backbone as inputs and concatenate them to output a single set of classes logits.

The resulting model contains 358 layers, with 28 260 266 parameters. The training process and the loss are the same as for the single domain classifier (equations (5) and (6)).

5.3.4 Supervised and Unsupervised training

Adversarial domain adaptation methods described above can be used in both supervised, semi-supervised and unsupervised setups.

In the supervised setup, both domains examples are processed throughout the entire model. Only the domain adaptive subnetwork distinguishes between source and target examples to compute the domain adaptation loss. Moreover, backbone and Head of the models are initialized with weights of the YOLOv8m model trained on the MS COCO dataset, while the new weights introduced by the domain adaptive subnetworks are initialized randomly.

In the unsupervised setup, only bird annotations from the source domains are used during training. Hence, the backbone of the YOLOv8 model processes images of both domains, and feeds corresponding feature maps to the domain classifier. However, only features corresponding to sources examples enter the detection head that output bounding boxes and classes and compute the detection losses. The model is then optimized by

back propagation of the sum of the detection losses between ground truths and models predictions on source examples and the domain classification loss computed on both source and target examples. The last epoch, reached after convergence of all losses is selected for the model evaluation and inference, in order to ensure the domain adaptive subnetwork convergence. To simulate real-life situations and study transfer learning possibilities, the model is initialized with the weights of the YOLOv8 detector trained on the source domain. Only the weights of the domain adaptive subnetwork are initialized randomly.

5.4 Evaluation process

In order to evaluate performances of the implemented architectures, the trained models are used in inference mode to detect birds on the test sets. A post-processing step is applied on the bounding boxes predicted by the models. First, low confidence bounding boxes are eliminated according to a confidence threshold. Then, Non-Maximum Suppression (NMS, [54]) is applied to eliminate possible duplicates, using a NMS IoU threshold. Final predictions for multiple confidence and IoU thresholds are studied to determine the best thresholds pair.

Finally, for each image, predictions are matched to the ground truths using an IoU threshold of 0.1. This low threshold accounts for the task's emphasis on detection counts, rather than accuracy of bounding boxes locations. Models performances are then quantified and compared using the Precision and Recall metrics, Precision-Recall curve and the average precision metric. Precision or positive predictive value indicates the quality of the model's positive predictions by measuring the ratio of correct predictions over the total amount of predictions. Recall or sensitivity measures the capacity of the model to detect all actual birds using the ratio of correct predictions over the total amount of birds. The Precision-Recall curve show the trade-off between the two metrics. The average precision summarizes this trade-off across all confidence threshold values, allowing easier comparison of models. It corresponds to the area under the Precision-Recall curve or the mean of the Precision values weighted by the increase in Recall at each threshold. The computations of Recall, Precision, and average precision are given in equations (8) and (9).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{8}$$

$$\text{AP} = \sum_{n} (\text{Recall}_n - \text{Recall}_{n-1}) * \text{Precision}_n \tag{9}$$

where TP, FP and FN accounts for True Positive (number of correct bird detections), False Positive (number of incorrect detections) and False Negative (number of birds missed by the model) respectively. The three metrics are also used throughout the experiment to give insights on the models' achievements and failure cases. Precision_n and Recall_n are the Precision and Recall values at the n-th confidence threshold.

For all models, evaluation metrics are computed and analyzed across each independent dataset, and across source and target test sets separately.

6 Results

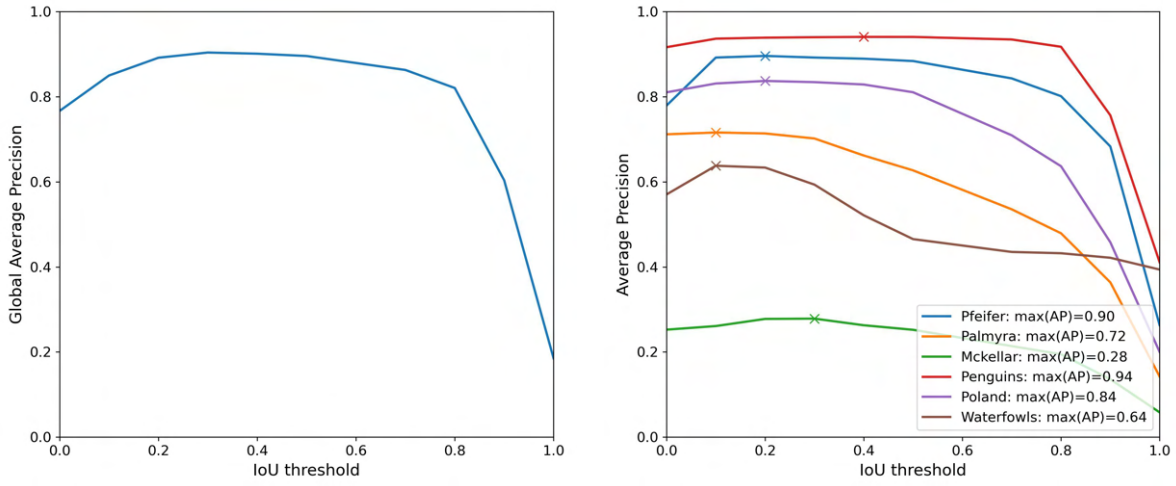
6.1 Baseline models

To understand the challenges of the study, a first model is trained on the global dataset. In the following of the experiment, this model is referred as the global model. Model parameters are determined using a grid search, and gathered in Table 9 of the Appendix.

IoU threshold study

The global model performances are studied across multiple IoU thresholds in Figure 9, highlighting some challenges due to the diversity of the global dataset. The left plot shows average precision values across the entire global dataset, while the right graph shows the metric values across distinct sources of the global dataset. For some sources, like the Terns dataset (Figure 2g), images present flocks of birds, very close and difficult to enumerate. Due to this particularity, bounding boxes annotations overlap considerably, and a high IoU threshold is necessary to ensure detection of all birds. This is illustrated on Figure 9b where smaller IoU thresholds considerably decrease average precision metric on the Terns dataset. On the other hand, other sources present partially visible birds, sliced by trees or other surrounding elements. This coupled with the varying size of birds across sources and the datasets imbalances leads the model to detect the same birds at multiple center locations and multiple scales. The model then present numerous unwanted bounding boxes overlaps. A smaller IoU threshold allows to get rid of most of these unwanted bounding boxes and considerably improve performances as illustrated by the Palmyra curve in Figure 9b.

The final choice of IoU threshold hence needs to balance these contradictory issues and illustrate a major challenge introduced by the variety of birds scales in the global dataset. Considering both graphs, following results and performances will be presented at an IoU threshold of 0.3 to maximize performances across each source.



(a) Average precision of the global model across multiple IoU thresholds.

(b) Average precision of the global model across IoU thresholds and sources. Points denote maximal AP values.

Figure 9: Grid search study for the Non Maximal Suppression IoU threshold. Average precision (AP) metric of the global model is computed across various IoU thresholds and distinct sources.

Figure 10 shows the average precision evaluated on the test set of each individual source, with the chosen IoU threshold of 0.3 and confidence threshold set to 0.1. Some prediction examples of the model are presented in Appendix B.2. The results reveal a disparity in performances across sources. Particularly, the Penguins, Terns and Hayes datasets show very high average precision values, even across IoU thresholds (see Figure 9b). On the other hand, the model has more struggles to detect birds on the McKellar, Palmyra and Waterfowls datasets.

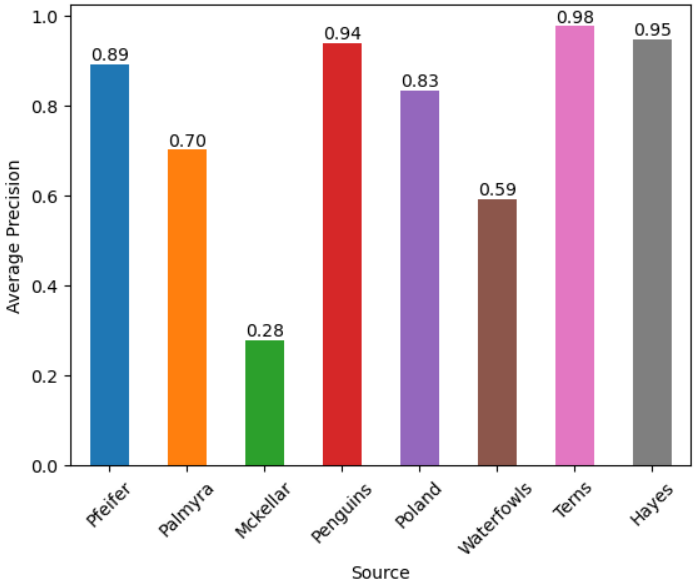
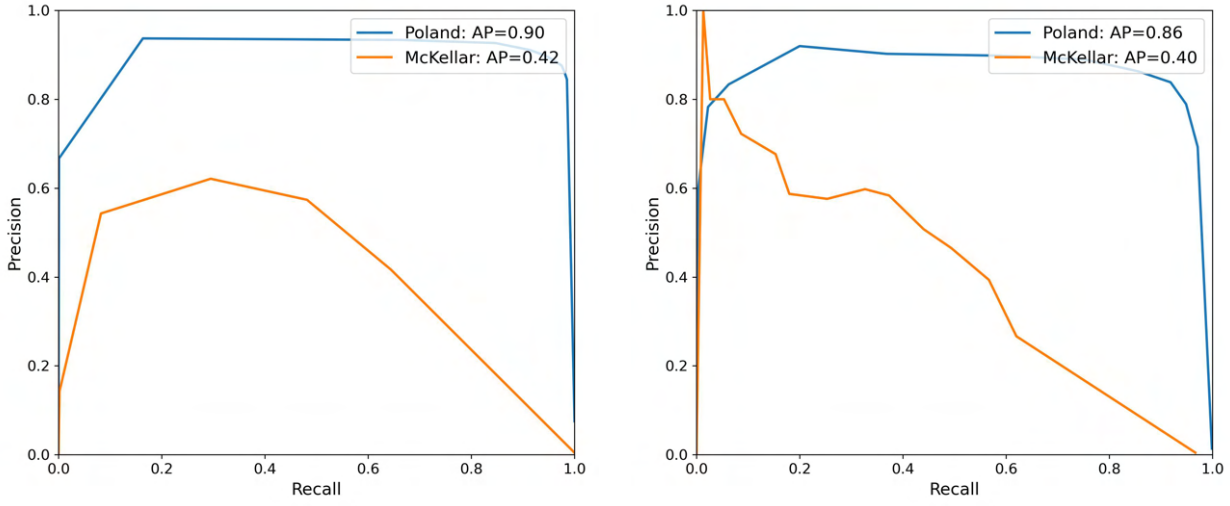


Figure 10: Average precision values of the global model trained on the global dataset training set and evaluated on each individual source testing set, using an IoU threshold of 0.3.

To explore the possibility to train an efficient model for multiple bird surveys, the following of the experiment focuses on two pairs of datasets *Penguins-Palmyra* and *Poland-McKellar*. In each pair, the two datasets present higher and lower average precision values in the global model. Figures 11a and 11b show the Precision-Recall curves of the YOLOv8 model trained on the two datasets of both pairs. Parameters used to train these two models and the following ones are provided in Table 9 in the Appendix. As visible, in both pairs of datasets studied, the model shows very good performances on one dataset (Penguins and Poland) but provides significantly lower average precision values on the second dataset (Palmyra and McKellar). For the first pair, the prediction examples provided in Figure 12 show the model misses most of the birds on the second dataset (only 2 out of the 11 birds present in the images are detected). Moreover, some darker elements are detected as birds, as illustrated by the second examples. Prediction examples for the McKellar dataset are given in Figure 29a in Appendix and show similar results.



(a) *Penguins-Palmyra* model

(b) *Poland-McKellar* model

Figure 11: Precison-Recall curves of the YOLOv8 model trained on multiple datasets combination. Training parameters are gathered in Table 9.

Figure 12: Prediction examples of the YOLOv8 model trained on *Penguins-Palmyra*. Ground truths are annotated in green and model predictions in red.

These results highlight the domain shift problem between datasets drawn from distinct probability distributions and presenting different ecosystems and species, but also sizes and resolutions. The following of the experiment studies multiple domain adaptation methods to address this issue.

6.2 Comparison of domain adaptation architectures in supervised setup

As mentioned previously, the domain shift problem can be tackled through multiple domain adaptation methods. Features alignment through l^2-norm minimization and the integration of domain adaptive networks are tested on the identified pairs, *Penguins-Palmyra* with Penguins considered as the source domain and Palmyra as the target domain and *Poland-McKellar* with Poland as the source and McKellar as the target domain.

All the domain adaptation methods are first trained in a supervised setup. Grid searches are performed to determine gains of extra losses introduced. In order to compare model performances, other parameters are set to the values defined by optimization of the original YOLOv8m model on the same set of data (Table 9) and the YOLOv8 models trained on each and both domains are also evaluated. All evaluations are performed with IoU threshold set to 0.3 and confidence threshold set to 0.1. Gird searches of losses weights and losses analysis of the distinct architectures are provided in Appendix B.3 for completeness.

6.2.1 Penguins-Palmyra

Performances of domain adaptation methods trained on the *Penguins* to *Palmyra* dataset combination are gathered in Table 3. The average precision metric show that all domain adaptation architectures outperforms original YOLOv8 models. When compared to the YOLOv8 model trained on target domain, the L2-norm minimization method already leads to a significant improvement of more than 50% of average precision on the target domain, demonstrating the possibility for the model to learn effective features for detection in both domains. The addition of domain classifiers further improves the results. The Multi-Domain Classifiers architecture presents the best results on target domain with an average precision value of 0.83, almost doubling performances of the baseline YOLOv8 model trained on both domains. Moreover, domain adaptation architectures also present slightly improved performances on the source domain (from 3.3 to 5.6% improvement), except for the Multi-Features Domain Classifier. The latter architecture leads to increased performances of around 69% on the target domain but presents a significant decrease of 10% on the source domain.

Model architectures		**AP on source**	**AP on target**
YOLOv8 models	trained on source	0.89	0.00
	trained on target	0.85	0.45
	trained on both domains	0.9	0.42
Domain Adaptation architectures	L2-norm	0.93	0.68
	Single Domain Classifier	**0.94**	0.71
	Multi-Domain Classifiers	0.92	**0.83**
	Multi-Features-Domain Classifier	0.80	0.76

Table 3: Comparison table of domain adaptation methods for adaptation from *Penguins* to *Palmyra*. Models are trained in a supervised setup and evaluated using average precision (AP) on source and target test sets separately.

(a) Prediction examples of the Features Alignment using L2-norm architecture.

(b) Prediction examples of the Single Domain Classifier architecture.

(c) Prediction examples of the Multi-Domain Classifiers architecture.

(d) Prediction examples of the Multi-Features Domain Classifier architecture.

Figure 13: Prediction examples of the domain adaptation architectures for adaptation from *Penguins* to *Palmyra*. Models are trained in a supervised setup and applied on the target test set. Groundtruths are in green and predictions are in red.

Figure 13 shows some prediction examples on the target domain. For comparison matters, predictions are shown on the same images for all models. On these examples, all domain adaptation methods succeed in detecting more birds than the YOLOv8 model (Figure 11) with the best architecture in term of average precision metric, the Multi-Domain Classifiers detecting 8 out of the 11 annotated birds. To gain a better understanding of these improvements, the True Positive, False Positive and False Negative metrics are reported in Table 4 for each architecture. Almost all domain adaptation architectures provide increased True Positive

and decreased False Positive and False Negative metrics. In the baseline YOLOv8 model, more than half of the birds present in the target domain are missed by the model while more than 40% of the model's predictions are incorrect. The domain adaptation methods significantly reduces these incorrect and missed detections, improving both the Recall and Precision metrics, as illustrated in Figure 14. Only the multi-domain classifiers architecture leads to an increase in False Positive, while significantly reducing the number of False Negative predictions.

Model architectures		TP	FP	FN
YOLOv8 trained on both domains		222	165	239
Domain Adaptation architectures	L2-norm min	322	129	139
	single Domain Classifier	353	128	108
	multi-Domain Classifiers	412	224	49
	multi-Features Domain Classifier	321	105	140

Table 4: True Positive (TP), False Positive (FP) and False Negative (FN) metrics of the domain adaptation architectures for adaptation from *Penguins* to *Palmyra*. Models are trained in a supervised setup and evaluated on the target domain test set.

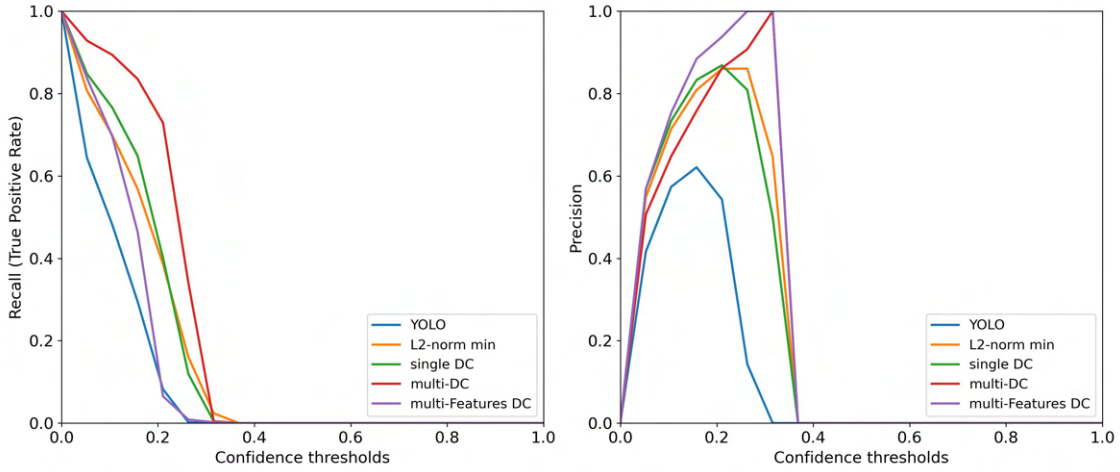


Figure 14: Recall (left) and Precision (right) curves of the domain adaptation models for adaptation from *Penguins* to *Palmyra*. Models are trained in a supervised setup. Recall and Precision metrics are evaluated on the target domain test set.

6.2.2 Poland-McKellar

Performances of the domain adaptation methods on the second pair of domains *Poland* to *McKellar* are gathered in Table 5. In this case, the domain adaptation methods also outperform the YOLOv8 model trained on both domains but do not reach performances of the YOLOv8 model trained on the target domain only. Moreover, in this case the increases of average precision values are smaller, going from 5% to 35% improvement on the target domain, when compared to the YOLOv8 model trained on the two domains. The higher average precision metrics on both source and target domains are obtained with the Multi-Features Domain Classifier architecture.

Model architectures		AP on source	AP on target
YOLOv8 models	trained on source	0.83	0.20
	trained on target	0.15	0.64
	trained on both domains	0.86	0.4
Domain Adaptation architectures	L2-norm	0.79	0.42
	Single Domain Classifier	0.86	0.52
	Multi-Domain Classifiers	0.93	0.51
	Multi-Features-Domain Classifier	**0.93**	**0.54**

Table 5: Comparison table of domain adaptation methods for adaptation from *Poland* to *McKellar*. Models are trained in a supervised setup and evaluated using average precision (AP) on source and target test sets.

The Precision-Recall curves on both domain, presented in Figure 15 as well as the True Positive, False Positive and False Negative metrics gathered in Table 6 confirm slight improvements of the domain adaptation methods. Although most domain adaptation architectures are not able to increase the number of True Positive, nor reduce the number of False Negative, they all lead to a decrease in number of False Positive. This last one is particularly high in the YOLOv8 baseline model, False Positive predictions representing more than 60% of the model's predictions. With the L2-norm minimization and the Multi-Features Domain Classifier architectures, the decrease is particularly important, False Positive predictions are reduced by more than 50% and 70% respectively, showing that the domain adaptation methods succeed in reducing the number of incorrect predicitons encountered in the baseline model. Some prediction examples are given in Figure 29 in Appendix.

Model architectures		TP	FP	FN
YOLOv8 trained on both domains		85	131	65
Domain Adaptation architectures	L2-norm	57	58	93
	single Domain Classifier	90	125	60
	multi-Domain Classifiers	80	121	70
	multi-Features Domain Classifier	69	30	81

Table 6: True Positive (TP), False Positive (FP) and False Negative (FN) metrics of the domain adaptation architectures for adaptation from *Poland* to *McKellar*. Models are trained in a supervised setup and evaluated on the target domain test set.

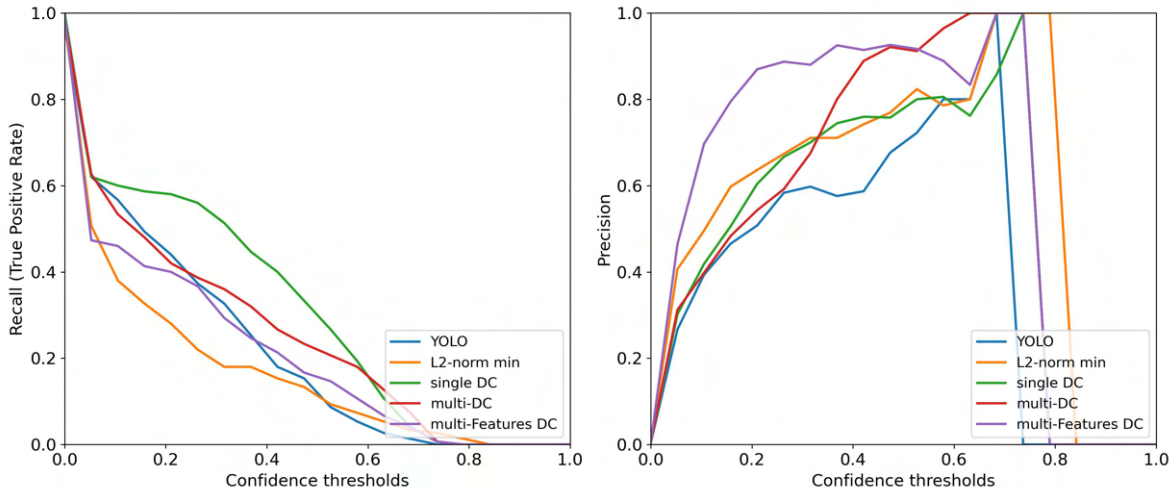


Figure 15: Recall (left) and Precision (right) curves of the domain adaptation models for adaptation from *Poland* to *McKellar*. Models are trained in a supervised setup. Recall and Precision are evaluated on the target domain test set.

6.3 Unsupervised domain adaptation

In order to evaluate the possibility to use domain adaptation methods on new unlabeled bird surveys, the methods implemented are tested in an unsupervised setup. Both source and target domains samples are used to compute the domain adaptive network loss while only source samples are considered in the bounding boxes regression and classification losses.

Table 7 gathers performances of the distinct domain adaptation architectures on the same dataset combinations studied previously, *Penguins* to *Palmyra*, and *Poland* to *McKellar*. In both cases, the domain adaptation architectures are initialized with the weights of the YOLOv8 model trained on the source dataset. This setup simulates the transfer learning scenario from an existing bird detector to a new unlabeled set of data. Like previously, the resulted models are evaluated on the two domains test sets, using average precision metric with an IoU threshold of 0.3.

Model architectures		*Penguins* to *Palmyra*		*Poland* to *McKellar*	
		AP on source	**AP on target**	**AP on source**	**AP on target**
YOLOv8 trained on source		0.91	0.01	0.83	0.20
Domain Adaptation architectures	L2-norm	0.92	0.01	0.94	0.25
	Single DC	0.90	**0.02**	**0.94**	0.34
	Multi-DC	0.88	0.01	0.93	0.37
	Multi-Features DC	**0.94**	0.01	0.93	**0.41**

Table 7: Comparison table of domain adaptation methods trained in the unsupervised setup. Results are given for adaptation from *Penguins* to *Palmyra* and *Poland* to *McKellar*. The average precision (AP) on each domain test set are reported separately.

Table 7 shows the domain adaptation architectures do not lead to significant improvement on the first target domain test set. In the second case however, the domain adaptation methods outperform the source model (YOLOv8 model trained on source domain). The best architecture, the Multi-Features Domain Classifier doubles the performances in term of average precision metric. These improvements are confirmed by observation of the True Positive, False Positive and False Negative metrics gathered in Table 8. Although the True Positive predictions slightly decrease in the domain adaptation architectures, the number of False Positive is significantly reduced, leading to a significant improvement in Precision as visible in Figure 16. Some prediction examples are provided in Figure 17.

Model architectures		**TP**	**FP**	**FN**
YOLOv8 trained on source		59	358	91
Domain Adaptation architectures	L2-norm	41	147	109
	single Domain Classifier	37	22	113
	multi-Domain Classifiers	39	26	111
	multi-Features Domain Classifier	51	20	99

Table 8: True Positive (TP), False Positive (FP) and False Negative (FN) metrics of the domain adaptation architectures from *Poland* to *McKellar*. Models are trained in the supervised setup and evaluated on the target domain test set.

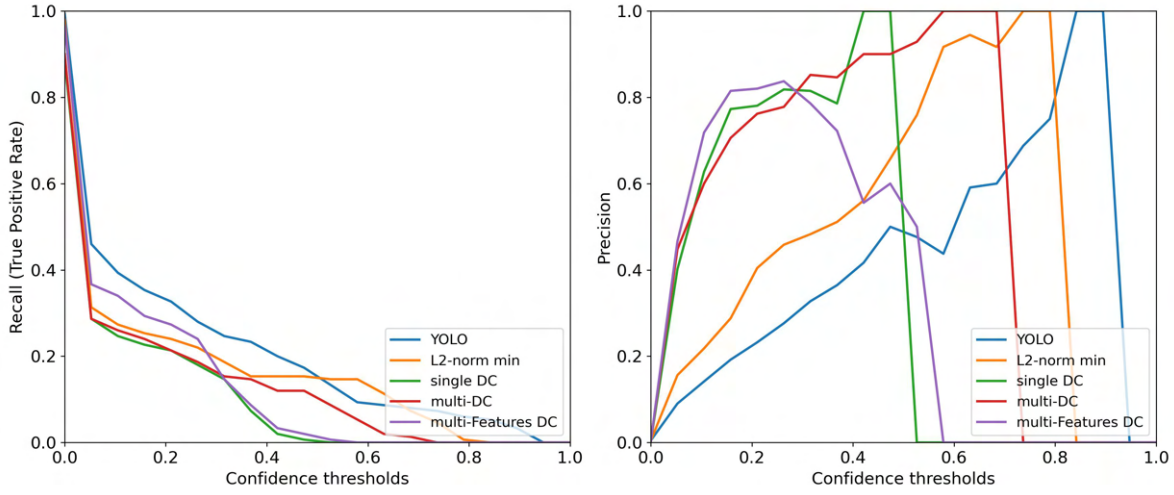
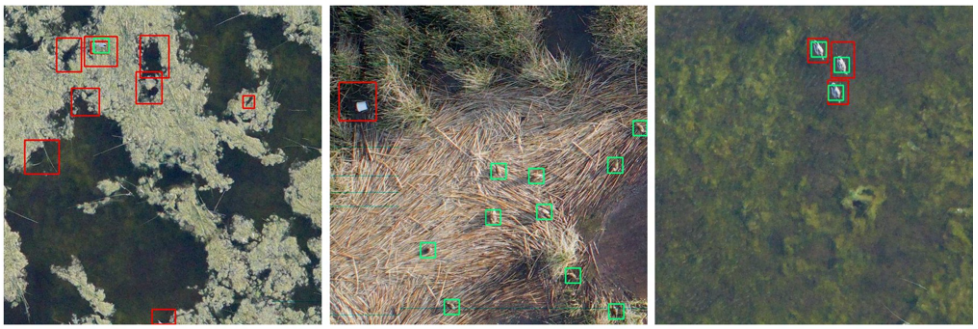
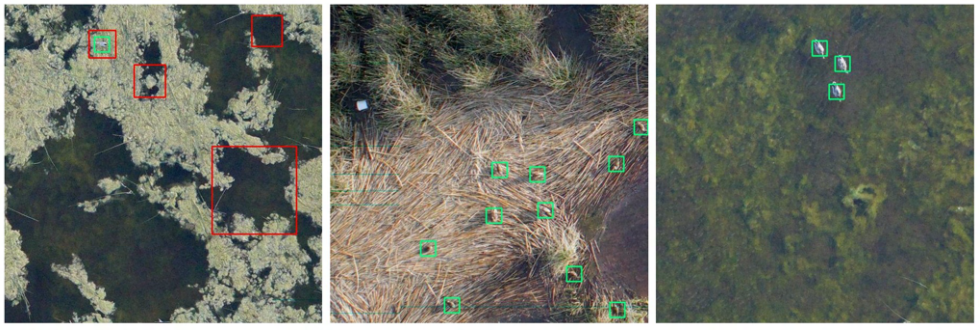


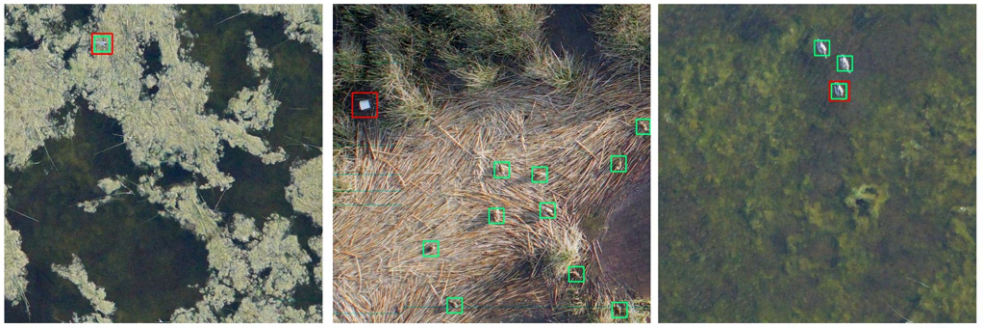
Figure 16: Recall (left) and Precision (right) curves of the domain adaptation methods for adaptation from *Poland* to *McKellar*. Models are trained in the unsupervised setup. Recall and Precision metrics are evaluated on the target domain test set.



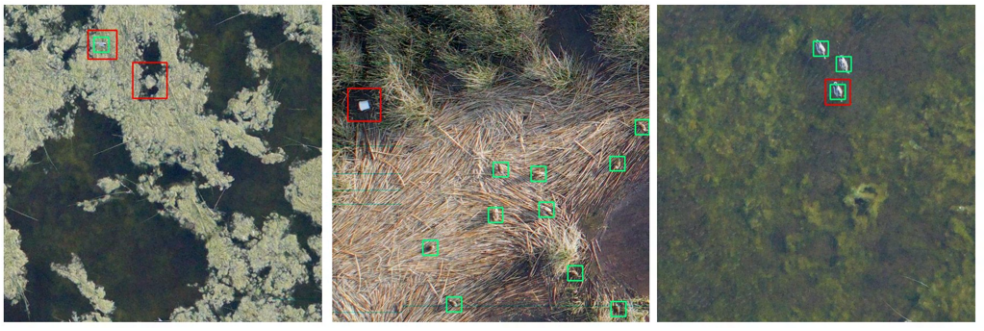
(a) Prediction examples of the YOLOv8 model.



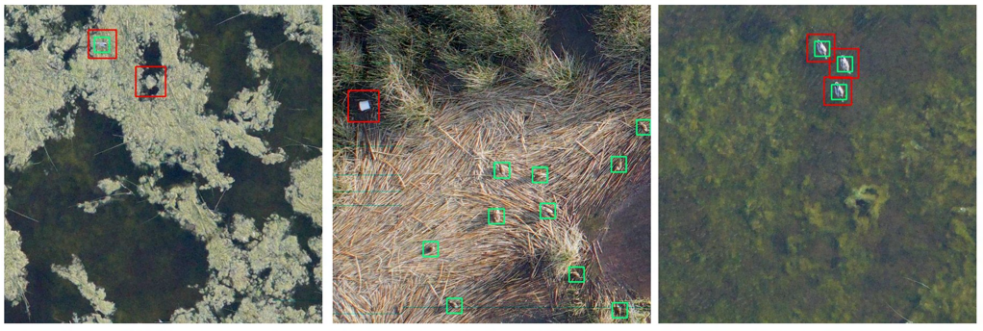
(b) Prediction examples of the Features Alignment using L2-norm architecture.



(c) Prediction examples of the Single Domain Classifier architecture.



(d) Prediction examples of the Multi-Domain Classifiers architecture.



(e) Prediction examples of the Multi-Features Domain Classifier architecture.

Figure 17: Prediction examples of the domain adaptation architectures from *Poland* to *McKellar*. Models are trained in the unsupervised setup and applied on the target test set. Groundtruths are in green and predictions are in red.

7 Discussion

7.1 Datasets

As mentioned before, one important complexity of the task of building a global and resolution-invariant bird detector relies on the creation of a global dataset. The combination of multiple surveys focusing on various species and ecosystems and with multiple resolutions, leads to very unbalanced datasets which impacts model performances on distinct sources. However, building a more balanced dataset is unrealistic as it would need to account for too many image characteristics to reflect the variety of surveys.

The baseline YOLOv8 models reveal that the amount and various characteristics of the datasets combined to train the models significantly impact its performances on individual datasets. The disparities in performances across individual sources of the global model can be explained by the dataset imbalances (in terms of both images and birds counts), as well as the characteristics of the domain shifts between individual sources. These disparities highlight the difficulty to create an efficient global dataset for generalized bird detectors. The issue was already encountered and reported by Weinstein et al. [81]. Their implemented general bird detector provides significantly higher performances (in terms of F1-score) on individual datasets visually closer to the larger dataset introduced in the training set, while presenting significantly lower performances on multiple individual datasets visually less similar. Like our global model, the generalized detector significantly suffers from dataset imbalances within the global training set.

A few failure cases of the global model can be identified. First, the large range of bird sizes within the global dataset adds a major difficulty. The differences in bird scales due to the camera settings, difference in altitudes but also species characteristics result in data with very different annotation sizes as visible in Table 2. Even when applying a scaling augmentation to simulate various sizes within each source, the global model often fails to distinguish between neighboring birds and larger birds. Larger birds are often detected as multiple birds, while some neighboring birds are often detected as unique birds. This is particularly visible in images of the Palmyra dataset, that present various sizes of birds (Figures 21c).
Furthermore, probably due to the high color contrast of birds and backgrounds in some of the datasets (birds are generally darker than the background in datasets like Penguins or Pfeifer, and very light in datasets like Poland), many False Positive errors occur on contrastive elements (darker rocks, shadows in trees, ...). This kind of error is particularly present in the McKellar dataset and can be observed on the first example in Figure 21e, but also in Figures 21g and 21h.

Finally, the combination of datasets makes the optimization of models more challenging. Indeed, the fitness metric used for selection of the best epoch and for grid search relies on performance evaluation on the global dataset. Without distinguishing between distinct sources, the fitness metric may suffer from dataset imbalances and favour larger datasets, biasing results. Introducing a more balanced validation metric however seems more challenging as it would increase computational complexity and requires precise knowledge about causes impacting datasets imbalances. Hence, verifying losses convergence at best epoch is very important in the actual setup, especially as we introduce the domain adaptive subnetworks.

7.2 Domain Adaptation architectures and supervised training

The major motivation for a generalized bird detector relies in the ability of such system to provide satisfying detections on new surveys, presenting all types of characteristics, sizes, ecosystems and species. Assessing the difficulties brought by the variety of data to build such generalized detector, a study on adversarial domain adaptation methods is conducted to resolve the domain shift problem when confronted to new datasets. The comparison of adaptation architectures with original YOLOv8 models in the supervised setup confirms implemented methods help the models provide better detections on the target domain test set for the two pairs of domains studied.

The features alignment method based on the minimization of l^2-norm between source and target features led to improved performances on the tested target domains and provides evidence for domain adaptation efficiency through features alignment. The architectures based on the addition of domain classifiers presented even greater improvements. In particular, taking advantage of the three scaled feature maps led the model to learn domain invariant features from earlier layers in the backbone. Inspired by prior works [30], we conducted further studies on the domain adaptation network architecture. Unlike expectations, the new architecture based on concatenation of multiple scale feature maps in a single domain classifier did not outperform the multi-domain classifiers architecture in both cases. If this architecture provides the higher average precision values

on the second pair of domains, it is not the most efficient one for the first pair. This may be due to differences in the two domain shifts studied. Moreover, it is important to notice that this last architecture presents particularly unstable performances for the first domain pair: it is very sensitive to the definition of the loss weights, providing very distinct performances for distinct values of the domain adaptation loss weight but also across distinct training runs. Thus, it was difficult to determine the best parameter.

For both pairs of domains studied, employed domain adaptation methods all outperformed the baseline original YOLOv8 model trained on both source and target domains. However, the causes and types of improvements obtained are different according to the domains tested. For the first domain pair, domain adaptation methods decreased both the number of missed birds and the number of incorrect detections, addressing two important issues of the baseline model. In the experiment on the second domain pair, the L2-norm minimization and multi-Features Domain Classifier architectures improvements mainly result from a significant decrease in the number of incorrect predictions which was a particularly important issue in the baseline YOLOv8 model.

Hence, for both pairs, observed improvements demonstrate domain adaptation architectures helped to overcome the imbalance between the source and target datasets (more annotations in the Penguins and Poland datasets) and lack of data in the target domain, but also some specific failure cases due to each domain shift encountered. However, adaptation models did not overcome all failure cases. In the first domain pair studied, most architectures struggle to properly distinguish between larger and adjacent birds as visible on the first example of Figures 13a, 13b, 13c and 13d, presenting no to small improvement when compared to the YOLOv8 model. Like in the global model, the scaling data augmentation did not succeed in solving this issue. Moreover, the decrease of False Positive predictions in the second pair comes with a noticeable increase of False Negative and thus deterioration in Recall.

7.3 Unsupervised Domain Adaptation for bird detection

Although the study of domain adaptation architectures in the supervised setup showed interesting improvements on both domain shifts studied, the results in the unsupervised case show more discrepancies (Table 7). The comparison of architectures performances on the first domain pair reveals domain adaptation methods fail to take advantage of knowledge learnt in the source domain to learn to detect birds in the target domain. On the other hand, when applied on the second set of domains, employed domain adaptation methods lead to significant improvements on the target test set. The best strategy (multi-Features Domain Classifier) doubles the average precision value on the target test set when compared to the source model (YOLOv8 trained on source). Like in the supervised setup, those improvements are mainly due to the decrease of False Positive, the domain adaptation models still missing a significant amount of birds.

These discrepancies in results highlight the differences in the domain shifts studied. Domain shifts are very difficult to quantify. Hence, it is complex to anticipate how much a model may suffer from the domain shift problem and how much domain adaptation methods may improve performances on target domains. Moreover, most previous works introducing methods for DAOD ([30], [80], [10]) focus and assess improvements on a few popular domain shifts, defined through common datasets. Some recurrent domain shifts studied are meteorological conditions with the Cityscapes [12] and Foggy Cityscapes [68] datasets, real and synthetic driving scenes (KITTI [18] and SIM 10K [33]), or other real and artistic scenes (PASCAL VOC [14] and Clipart [55]). It is challenging to assess similarities and disparities between the domain shifts encountered in this study and the one studied in most DAOD methods.

Several works introduced metrics to quantify domain shifts between datasets, and predict drops in model performances [68], [52], [35]. For image datasets, Kalogeiton et al. [35] proposed four domain shift factors (spatial location accuracy, appearance diversity, image quality and aspect distribution) impacting detectors performances. However, like highlighted in corresponding works, most metrics are still unable to account for the whole performance gap between source and target domains, highlighting the difficulty to identify all domain shift causes and characteristics.

In the domain shifts investigated in this study, the performance of the source model (original detector trained on source) applied on the target domain gives interesting insights. In the first domain shift study, when the source model is not able to detect any birds of the target dataset, domain adaptation methods did not improve detection performances. On the other hand, in the case of the *Poland-McKellar* experiment, the source model was already able to detect a few birds in the McKellar dataset and the domain adaptation methods significantly improved performances. This reveals that the domain shift in the first experiment may be too large to be addressed using the adversarial domain adaptation methods tested in this study. In the second domain shift, source and target features are close enough in the feature space for the domain adaptation strategies to

build domain invariant feature maps.

Moreover, the data investigation in section 3 also gives valuable insights for the characterization of the domain shifts studied. The imbalances in number of annotations between the source and target dataset (12003 and 3194 annotations respectively), as well as the mean number of birds per patch (12.77 and 1.83) reveal a non negligible domain shift in the label space in the *Penguins-Palmyra* case. On the contrary, the Poland and McKellar datasets present smaller differences in number of images and annotations (946 images and 4811 annotations in the Poland dataset, against 974 images and 2329 annotations in McKellar). These discrepancies in the label space domain shift can also explain the differences in domain adaptation results.

Finally, if the domain adaptation methods provide significant improvement on the McKellar target dataset, Recall remains low and performances do not reach the ones of the YOLOv8 model trained on the target domain only (average precision of 0.68, see Table 5). Chen et al. [10] and Zhang et al. [85] introduced an additional instance-level features alignment step, tailored for detection task. Considering the importance of the domain shift between current datasets, leading the network to focus on instance-level alignment could improve performances. It is noteworthy that in the current YOLOv8 architecture, unsupervised instance-level feature alignment would rely on the head's predicted bounding boxes, which are extremely numerous due to the high number of anchors used for bounding boxes proposals. Such domain adaptation method would then considerably increase the computational complexity in terms of storage. Other options to further explore domain adaptation possibilities on both domain shifts studied include generative domain adversarial adaptation that create target-like unreal samples using Generative Adversarial Network (GAN [21]) [6], or the integration of a student-teacher framework, using pseudo-labels or generative networks to adjust the framework for unsupervised domain adaptation [13], [88]. Considering improvements of the employed domain adaptation methods, a combination of these methods with the student-teacher framework, like in Li et al. [43] and Xiao et al. [84] could be promising.

Using semi-supervised learning by incorporating a few target labeled examples in the training process could also be a strategy to improve adaptation results, especially in larger domain shift like the first one studied, while still limiting the high costs of labeling processes. The improvements observed on both domain shifts in the supervised setup support this possibility. However, the quantity of labels required to obtain satisfying results needs to be verified and taken into account to assess the viability of such method. Moreover, semi-supervised learning could enhance possible shifts within the target domain itself and results may be impacted by the choice of labels. Indeed, aerial images within a single dataset often present wide variations making the adaptation task even more challenging. Particularly, Figure 2b illustrates the Palmyra images show a large variety of backgrounds which can create significant discrepancies across train, validation and test set. This can also explain the bad performances of domain adaptation architectures on the target test set of the first domains pair.

8 Conclusion

Motivated by the willingness to ease birds surveys using aerial images, this study exposed challenges brought by the construction of a generalized model for bird detection and explored possibilities offered by adversarial domain adaptation through the implementation of domain classifiers.

Using birds surveys gathered from multiple previous works and the recent state-of-the-art detection model YOLOv8, the analysis of a global model trained on multiple surveys highlighted the challenges to build an efficient dataset and train a generalized bird detector. Particularly, the analysis revealed large performance disparities due to imbalances and domain shifts between the distinct surveys. Focusing on two pairs of datasets, several experiments were conducted to address these domain shifts and explore the possibility to create an unsupervised transfer learning pipeline. Four distinct domain adaptation architectures were implemented and compared on two pairs of birds surveys, in both supervised and unsupervised setups.

Experiments in the supervised case showed promising results, revealing the possibility to use adversarial domain adaptation to improve model performances on smaller target datasets. Specifically, the implementation of domain classifiers taking advantage of the three scaled feature maps of the feature extractor showed the best performances in term of average precision metric. The domain adaptation strategies allowed the model to overcome challenges specific to the encountered domain shifts, decreasing significantly the number of missed instances in one case and reducing the important number of incorrect predictions in the second domain shift studied. However, the models still suffers from some noticeable failure cases introduced by the variety of data like the distinction between smaller neighboring birds and larger birds.

The study of domain adaptation methods in the unsupervised setup revealed both the challenges brought by domain adaptation for detection task and the difficulty to quantify domain shifts. The proposed architectures demonstrated domain adaptation possibilities on smaller domain shifts, showing improved performances on one of the unlabeled target birds surveys. The experiment also suggested the evaluation of performances of the original model used for transfer learning can provide insights on both domain shift characterization and possible improvements of domain adaptation methods. Those results are promising and demonstrate the possibility to use domain adaptation to create an efficient transfer learning pipeline and limit labeling costs in the case of bird detection from aerial images. More domain adaptation strategies can be tested to further improve models performances and address larger domain shifts, like instance-level features alignment [10], generative adversarial domain adaptation [6], or the integration of domain classifiers strategies in a student-teacher network [43], [84].

Acknowledgements

I would like to thank Benjamin Kellenberger for his close supervision and support throughout my thesis. I also thank Devis Tuia for his supervision and feedbacks. I am thankful to both of them for introducing me to the topic of this thesis, on which I really enjoyed working.
Finally, I thank all the members of the Center for Biodiversity and Global Change for their warm welcome in the laboratory. I fully enjoyed my stay at Yale and am very grateful for this experience that helped me discover the beautiful world of research in ecology in such a nice environment.

References

[1] Abhijeet Awasthi and Sunita Sarawagi. Continual learning with neural networks: A review. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pages 362–365, 2019.

[2] Jayme Garcia Arnal Barbedo, Luciano Vieira Koenigkan, Thiago Teixeira Santos, and Patrícia Menezes Santos. A study on the detection of cattle in uav images using deep learning. *Sensors*, 19(24):5436, 2019.

[3] Sara Beery, Dan Morris, and Siyu Yang. Efficient Pipeline for Camera Trap Image Review.

[4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[5] Elizabeth Bondi, Debadeepta Dey, Ashish Kapoor, Jim Piavis, Shital Shah, Fei Fang, Bistra Dilkina, Robert Hannaford, Arvind Iyer, Lucas Joppa, et al. Airsim-w: A simulation environment for wildlife conservation with uavs. In *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*, pages 1–12, 2018.

[6] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017.

[7] Qi Cai, Yingwei Pan, Chong-Wah Ngo, Xinmei Tian, Lingyu Duan, and Ting Yao. Exploring object relation in mean teacher for cross-domain detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11457–11466, 2019.

[8] Estelle Chabanel. BirdDetector. https://github.com/EstelleChabanel/BirdDetector, 2023.

[9] Dominique Chabot and Charles M Francis. Computer-automated bird detection and counts in high-resolution aerial images: a review. *Journal of Field Ornithology*, 87(4):343–359, 2016.

[10] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3339–3348, 2018.

[11] Gong Cheng and Junwei Han. A survey on object detection in optical remote sensing images. *ISPRS journal of photogrammetry and remote sensing*, 117:11–28, 2016.

[12] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Scharwächter, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset. In *CVPR Workshop on the Future of Datasets in Vision*, volume 2. sn, 2015.

[13] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4091–4101, 2021.

[14] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111:98–136, 2015.

[15] Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia. A brief review of domain adaptation. *Advances in data science and information engineering: proceedings from ICDATA 2020 and IKE 2020*, pages 877–894, 2021.

[16] Peter Frederick, Dale E Gawlik, John C Ogden, Mark I Cook, and Michael Lusk. The white ibis and wood stork as indicators for restoration of the everglades ecosystem. *Ecological indicators*, 9(6):S83–S95, 2009.

[17] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of machine learning research*, 17(59):1–35, 2016.

[18] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. The kitti vision benchmark suite. *URL http://www. cvlibs. net/datasets/kitti*, 2(5), 2015.

[19] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[20] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[21] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[22] GoogleApis. Open Images Dataset V7 and Extensions. https://storage.googleapis.com/openimages/web/index.html, 2022. [Online; accessed 28-December-2023].

[23] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789–1819, 2021.

[24] Richard D Gregory and Arco van Strien. Wild bird indicators: using composite population trends of birds as measures of environmental health. *Ornithological Science*, 9(1):3–22, 2010.

[25] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. Cross modal distillation for supervision transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2827–2836, 2016.

[26] Mahta HassanPour Zonoozi and Vahid Seydi. A survey on adversarial domain adaptation. *Neural Processing Letters*, 55(3):2429–2469, 2023.

[27] Madeline C Hayes, Patrick C Gray, Guillermo Harris, Wade C Sedgwick, Vivon D Crawford, Natalie Chazal, Sarah Crofts, and David W Johnston. Drones and deep learning produce accurate and efficient monitoring of large-scale seabird colonies. *The Condor*, 123(3):duab022, 2021.

[28] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[29] Mazin Hnewa and Hayder Radha. Multiscale domain adaptive yolo for cross-domain object detection. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 3323–3327. IEEE, 2021.

[30] Mazin Hnewa and Hayder Radha. Integrated multiscale domain adaptive yolo. *IEEE Transactions on Image Processing*, 32:1857–1867, 2023.

[31] Suk-Ju Hong, Yunhyeok Han, Sang-Yeon Kim, Ah-Yeong Lee, and Ghiseok Kim. Application of deep-learning methods to bird detection using unmanned aerial vehicle imagery. *Sensors*, 19(7):1651, 2019.

[32] Bojana Ivosevic, Yong-Gu Han, Youngho Cho, and Ohseok Kwon. The use of conservation drones in ecology and wildlife research. *Journal of Ecology and Environment*, 38(1):113–118, 2015.

[33] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *arXiv preprint arXiv:1610.01983*, 2016.

[34] Krish Kabra, Alexander Xiong, Wenbin Li, Minxuan Luo, William Lu, Tianjiao Yu, Jiahui Yu, Dhananjay Singh, Raul Garcia, Maojie Tang, et al. Deep object detection for waterbird monitoring using aerial imagery. In *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 455–460. IEEE, 2022.

[35] Vicky Kalogeiton, Vittorio Ferrari, and Cordelia Schmid. Analysing domain shift factors between videos and images for object detection. *IEEE transactions on pattern analysis and machine intelligence*, 38(11):2327–2334, 2016.

[36] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4893–4902, 2019.

[37] Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. Generalization in deep learning. *arXiv preprint arXiv:1710.05468*, 1(8), 2017.

[38] Benjamin Kellenberger, Diego Marcos, and Devis Tuia. Detecting mammals in uav images: Best practices to address a substantially imbalanced dataset with deep learning. *Remote sensing of environment*, 216:139–153, 2018.

[39] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[40] Wouter M Kouw and Marco Loog. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*, 2018.

[41] Ce Li, Baochang Zhang, Hanwen Hu, and Jing Dai. Enhanced bird detection from low-resolution aerial image using deep neural networks. *Neural Processing Letters*, 49:1021–1039, 2019.

[42] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *Advances in Neural Information Processing Systems*, 33:21002–21012, 2020.

[43] Yu-Jhe Li, Xiaoliang Dai, Chih-Yao Ma, Yen-Cheng Liu, Kan Chen, Bichen Wu, Zijian He, Kris Kitani, and Peter Vajda. Cross-domain adaptive teacher for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7581–7590, 2022.

[44] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[45] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[46] Julie Linchant, Jonathan Lisein, Jean Semeki, Philippe Lejeune, and Cédric Vermeulen. Are unmanned aircraft systems (uas s) the future of wildlife monitoring? a review of accomplishments and challenges. *Mammal Review*, 45(4):239–252, 2015.

[47] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016.

[48] Yang Liu, Peng Sun, Max R Highsmith, Nickolas M Wergeles, Joel Sartwell, Andy Raedeke, Mary Mitchell, Heath Hagy, Andrew D Gilbert, Brian Lubinski, et al. Performance comparison of deep learning techniques for recognizing birds in aerial images. In *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pages 317–324. IEEE, 2018.

[49] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015.

[50] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[51] Frederic Maire, Luis Mejias Alvarez, and Amanda Hodgson. Automating marine mammal detection in aerial images captured during wildlife surveys: a deep learning approach. In *AI 2015: Advances in Artificial Intelligence: 28th Australasian Joint Conference, Canberra, ACT, Australia, November 30–December 4, 2015, Proceedings 28*, pages 379–385. Springer, 2015.

[52] Carianne Martinez, David A Najera-Flores, Adam R Brink, D Dane Quinn, Eleni Chatzi, and Stephanie Forrest. Confronting domain shift in trained neural networks. In *NeurIPS 2020 Workshop on Preregistration in Machine Learning*, pages 176–192. PMLR, 2021.

[53] Assefa M Melesse, Qihao Weng, Prasad S Thenkabail, and Gabriel B Senay. Remote sensing sensors and applications in environmental resources mapping and modelling. *Sensors*, 7(12):3209–3241, 2007.

[54] Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th international conference on pattern recognition (ICPR'06)*, volume 3, pages 850–855. IEEE, 2006.

[55] Li Niu, Wen Li, and Dong Xu. Visual recognition by learning from web data: A weakly supervised domain generalization approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2774–2783, 2015.

[56] Mohammad Sadegh Norouzzadeh, Dan Morris, Sara Beery, Neel Joshi, Nebojsa Jojic, and Jeff Clune. A deep active learning system for species identification and counting in camera trap images. *Methods in ecology and evolution*, 12(1):150–161, 2021.

[57] Mohammad Sadegh Norouzzadeh, Anh Nguyen, Margaret Kosmala, Alexandra Swanson, Meredith S Palmer, Craig Packer, and Jeff Clune. 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, 2018.

[58] Poojan Oza, Vishwanath A Sindagi, Vibashan Vishnukumar Sharmini, and Vishal M Patel. Unsupervised domain adaptation of object detectors: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[59] Taesung Park, Alexei A Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 319–345. Springer, 2020.

[60] B Prasanna, Sunandini Sanyal, and R Venkatesh Babu. Continual domain adaptation through pruning-aided domain-specific weight modulation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2457–2463. IEEE, 2023.

[61] RangeKing. Brief summary of YOLOv8 model structure 189. https://github.com/ultralytics/ultralytics/issues/189, 2023. [Online; accessed 28-December-2023].

[62] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[63] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[64] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[65] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[66] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[67] Nicolas Rey, Michele Volpi, Stéphane Joost, and Devis Tuia. Detecting animals in african savanna with uavs and the crowds. *Remote Sensing of Environment*, 200:341–351, 2017.

[68] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126:973–992, 2018.

[69] Stefan Schneider, Graham W Taylor, and Stefan Kremer. Deep learning object detection methods for ecological camera trap data. In *2018 15th Conference on computer and robot vision (CRV)*, pages 321–328. IEEE, 2018.

[70] Dino Sejdinovic, Bharath Sriperumbudur, Arthur Gretton, and Kenji Fukumizu. Equivalence of distance-based and rkhs-based statistics in hypothesis testing. *The annals of statistics*, pages 2263–2291, 2013.

[71] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[72] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. Incremental learning of object detectors without catastrophic forgetting. In *Proceedings of the IEEE international conference on computer vision*, pages 3400–3409, 2017.

[73] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14*, pages 443–450. Springer, 2016.

[74] Sanli Tang, Zhanzhan Cheng, Shiliang Pu, Dashan Guo, Yi Niu, and Fei Wu. Learning a domain classifier bank for unsupervised adaptive object detection. *arXiv preprint arXiv:2007.02595*, 2020.

[75] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017.

[76] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE international conference on computer vision*, pages 4068–4076, 2015.

[77] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104:154–171, 2013.

[78] Ultralytics. Ultralytics YOLOv8 Docs. https://docs.ultralytics.com/, 2023. [Online; accessed 28-December-2023].

[79] wangs311. awesome-domain-adaptation-object-detection . https://github.com/wangs311/awesome-domain-adaptation-object-detection, 2023. [Online; accessed 30-December-2023].

[80] Jian Wei, Qinzhao Wang, and Zixu Zhao. Yolo-g: Improved yolo for cross-domain object detection. *Plos one*, 18(9):e0291241, 2023.

[81] Ben G Weinstein, Lindsey Garner, Vienna R Saccomanno, Ashley Steinkraus, Andrew Ortega, Kristen Brush, Glenda Yenni, Ann E McKellar, Rowan Converse, Christopher D Lippitt, et al. A general deep learning model for bird detection in high-resolution airborne imagery. *Ecological Applications*, 32(8):e2694, 2022.

[82] Ken Whitehead and Chris H Hugenholtz. Remote sensing of the environment with small unmanned aircraft systems (uass), part 1: A review of progress and challenges. *Journal of Unmanned Vehicle Systems*, 2(3):69–85, 2014.

[83] Serge A Wich, Mike Hudson, Herizo Andrianandrasana, and Steven N Longmore. Drones for conservation. *Conservation technology*, 35, 2021.

[84] Ruixin Xiao, Zhilei Liu, and Baoyuan Wu. Teacher-student competition for unsupervised domain adaptation. In *2020 25th international conference on pattern recognition (ICPR)*, pages 8291–8298. IEEE, 2021.

[85] Shizhao Zhang, Hongya Tuo, Jian Hu, and Zhongliang Jing. Domain adaptive yolo for one-stage cross-domain detection. In *Asian Conference on Machine Learning*, pages 785–797. PMLR, 2021.

[86] Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 12993–13000, 2020.

[87] Zhaohui Zheng, Ping Wang, Dongwei Ren, Wei Liu, Rongguang Ye, Qinghua Hu, and Wangmeng Zuo. Enhancing geometric factors in model learning and inference for object detection and instance segmentation. *IEEE transactions on cybernetics*, 52(8):8574–8586, 2021.

[88] Huayi Zhou, Fei Jiang, and Hongtao Lu. Ssda-yolo: Semi-supervised domain adaptive yolo for cross-domain object detection. *Computer Vision and Image Understanding*, 229:103649, 2023.

Data sources

[Data1] Vienna R. Saccomanno Ashley Steinkraus Andrew Ortega Kristen Brush Glenda Yenni Ann E. McKellar Rowan Converse Christopher D. Lipitt Alex Wegmann Nick D. Holmes Alice J. Edney Tom Hart Mark J. Jessopp Rohan Clarke Dominik Markowski Henry Senyondo Ryan Dotson ... S.K Morgan Ernest. Ben Weinstein, Lindsey Garner. A global model of bird detection in high resolution airborne images using computer vision. https://doi.org/10.5281/zenodo.5033174, 2021. [Online; accessed 2-October-2023].

[Data2] Gray P. C. Harris G. Sedgwick W. C. Crawford V. D. Chazal N. Crofts S. Johnston D. W. Hayes, M. C. Data from: Drones and deep learning produce accurate and efficient monitoring of large-scale seabird colonies. Duke Research Data Repository. https://doi.org/10.7924/r4dn45v9g, 2020. [Online; accessed 2-October-2023].

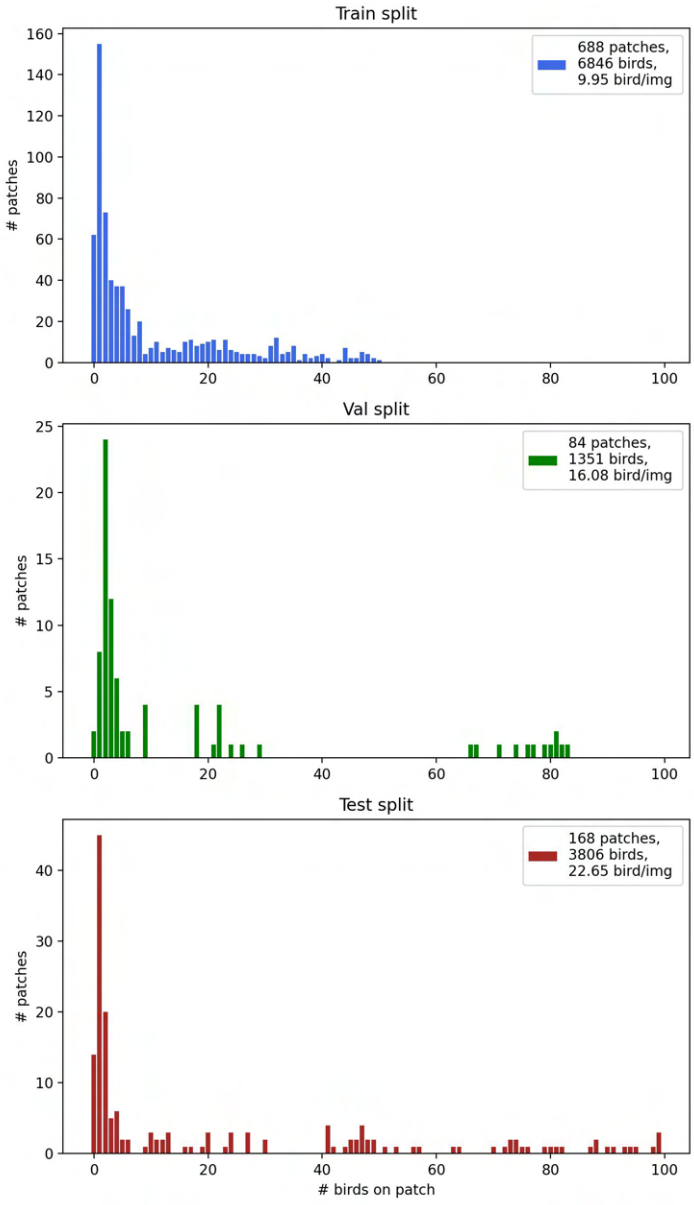
[Data3] Jacob; Woldt Wayne; Tang Zhenghong Hu, Qiao; Smith. UAV-derived waterfowl thermal imagery dataset, Mendeley Data, V4. https://doi.org/10.17632/46k66mz9sz.4, 2021. [Online; accessed 2-October-2023].

[Data4] Folmer E Tuia D. Kellenberger B, Veen T. 21,000 birds in 4.5 h: efficient large-scale seabird detection with machine learning. Remote Sensing in Ecology and Conservation. https://doi.org/10.1002/rse2.200, 2021. [Online; accessed 2-October-2023].

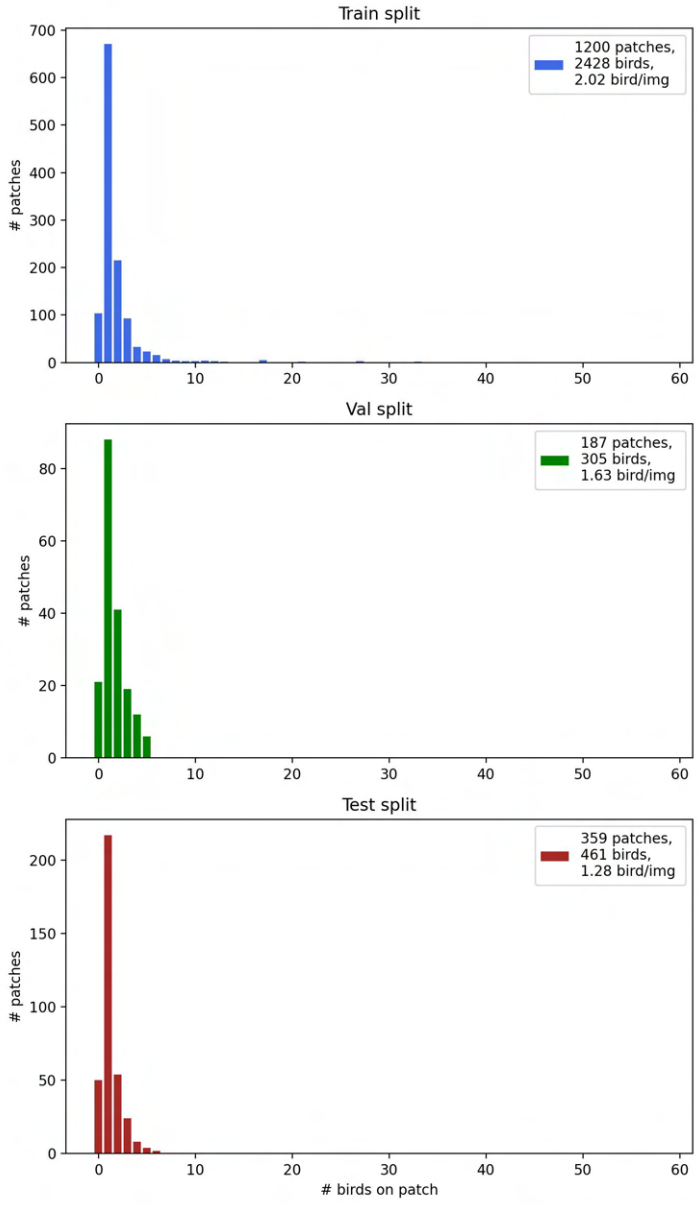
[Data5] Dan Morris. Datasets with annotated wildlife in drone/aerial images. https://github.com/agentmorris/agentmorrispublic/blob/main/drone-datasets.md, 2023. [Online; accessed 1-November-2023].

[Data6] Christian Pfeifer, Marie-Charlott Rümmler, and Osama Mustafa. Assessing colonies of antarctic shags by unmanned aerial vehicle (uav) at south shetland islands, antarctica. *Antarctic Science*, 33(2):133–149, 2021.

A Data

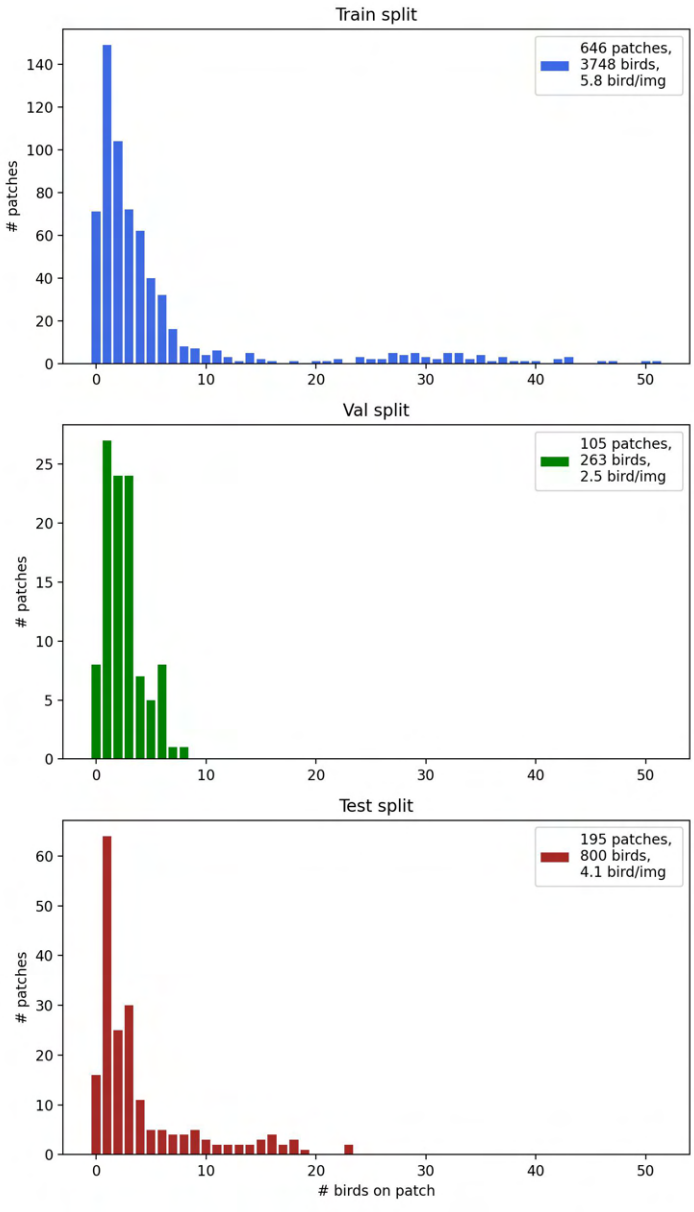


(a) Annotations distribution across train, validation and test sets of the Penguins dataset.

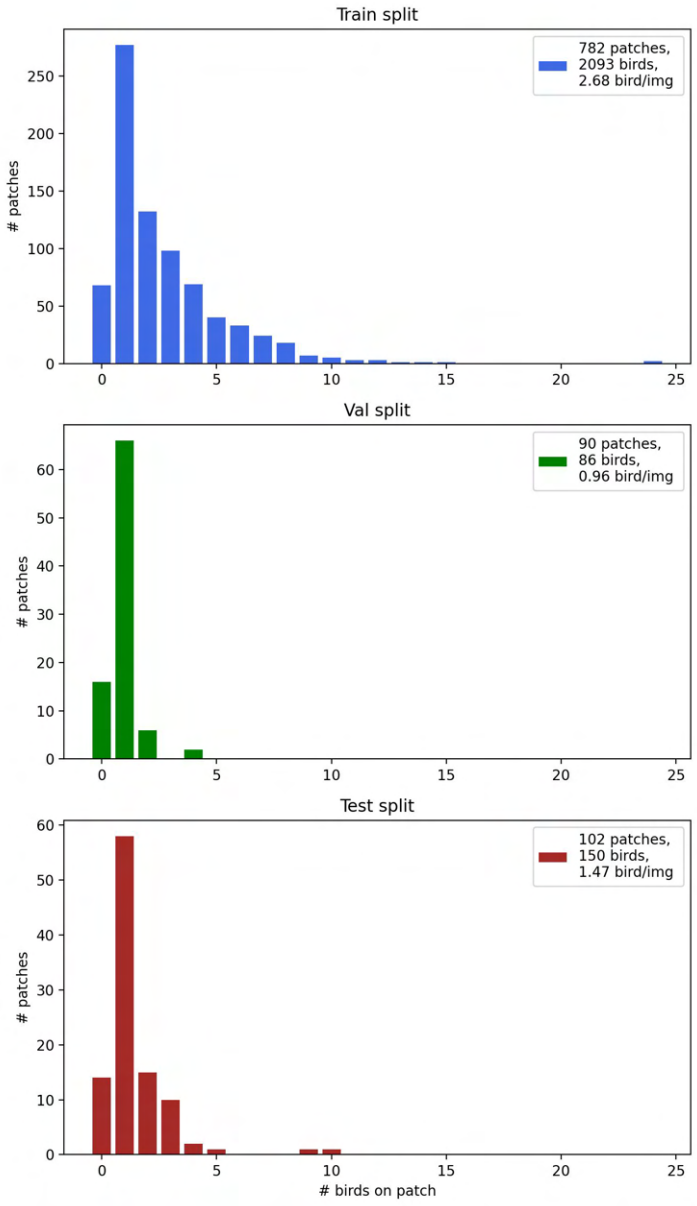


(b) Annotations distribution across train, validation and test sets of the Palmyra dataset.

Figure 18: Annotations distribution across train, validation and test sets of the Penguins and Palmyra datasets.



(a) Annotations distribution across train, validation and test sets of the Poland dataset.



(b) Annotations distribution across train, validation and test sets of the McKellar dataset.

Figure 19: Annotations distribution across train, validation and test sets of the Poland and McKellar datasets.

B Results

B.1 Grid search and training parameters

For each of the baseline models, training parameters gathered in Table 9 were defined following a grid search. Multiple combinations of learning rates, weight decay, momentum and augmentation parameters were tested along pre-defined range of values. Resulting models were compared using the same fitness metrics used by YOLOv8 to determine model weights. The optimizer Adam[39] and AdamW [50] were also tested for a few combination of parameters, presenting less accurate predictions than the SGD optimizer which is therefore kept for all the experiments.
Figure 20 shows an example of parameter tuning realised for the YOLOv8m model trained on the *Penguins-Palmyra* dataset.

Training parameters	**Global model**	**{*Penguins-Palmyra*} experiment**	**{*Poland-McKellar*} experiment**
epoch	200	200	200
patience	50	30	30
batch_size	32	32	32
optimizer	SGD	SGD	SGD
initial learning rate	0.01	0.00979	0.00691
final learning rate factor	0.01	0.01	0.01276
weight decay	0.0005	0.00048	0.00058
momentum	0.937	0.94971	0.937
warmup epochs	3.0	3.0	3.0
warmup momentum	0.8	0.8	0.8
box gain	7.5	8.23357	7.5
cls gain	0.5	0.53081	0.47651
dfl gain	1.5	1.72224	1.463
degrees	45	44.72491	44.82499
horizontal flip probability	0.5	0.48174	0.58144
vertical flip probability	0.5	0.47555	0.58409
scale	0.5	0.47999	0.4757
hsv_h	0.0	0.0146	0.01499
hsv_s	0.0	0.7305	0.7
hsv_v	0.0	0.39509	0.30742
L2-norm min loss weight	-	0.25	0.25
single DC loss weight	-	1.5	1.0
multi-DC losses weights	-	0.5, 0.5, 1.0,	1.0, 1.0, 1.0
multi-features DC loss weight	-	0.5	1.5

Table 9: Training parameters of the different models studied. hsv_h, hsv_s and hsv_v respectively denotes the hue, saturation and values augmentation ratios. Last 4 parameters correspond to losses weights of the distinct domain adaptation architectures tested.

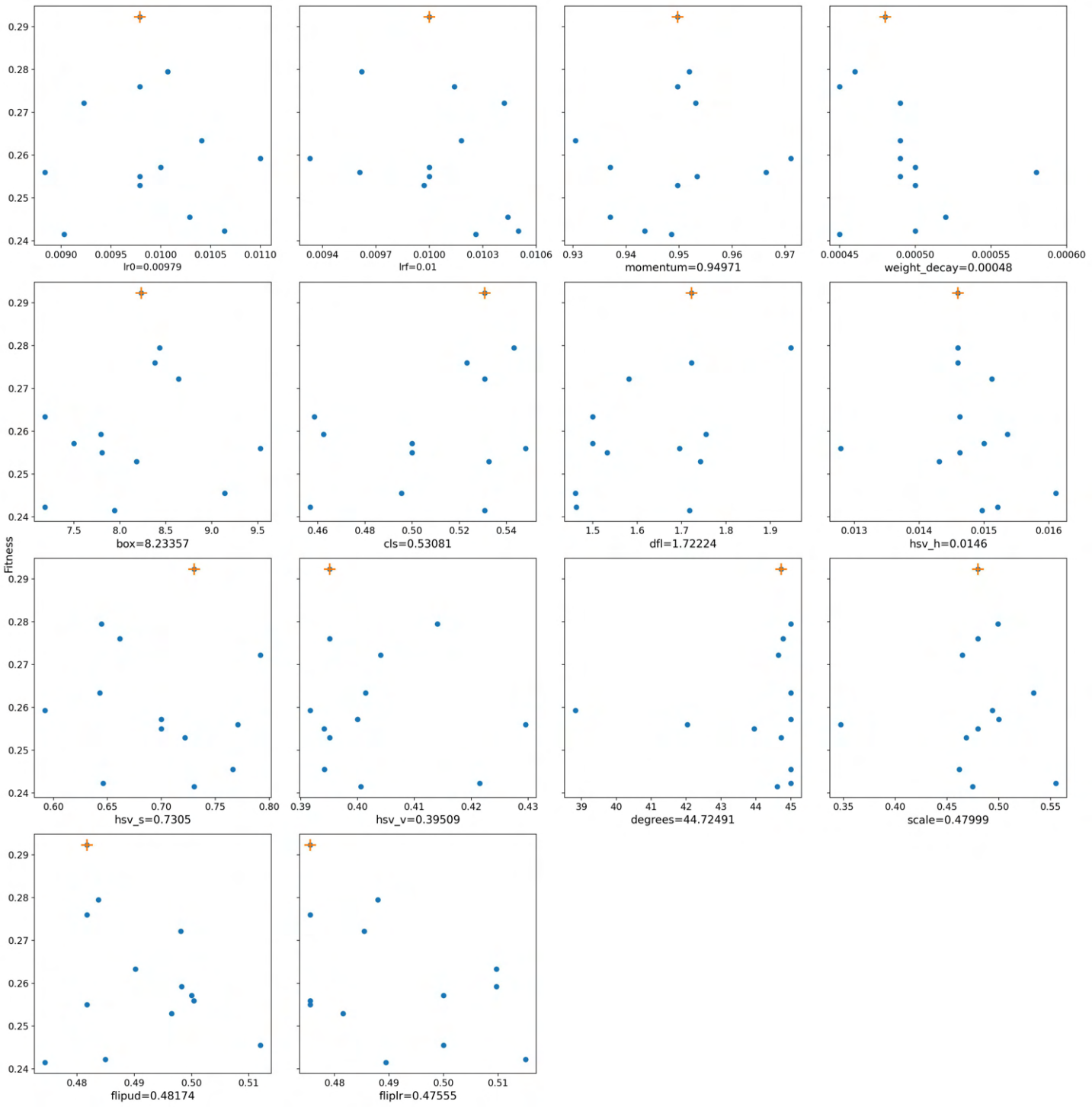
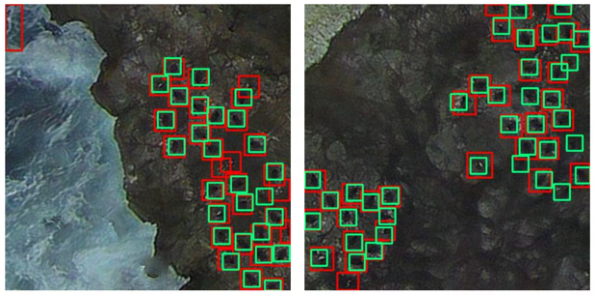
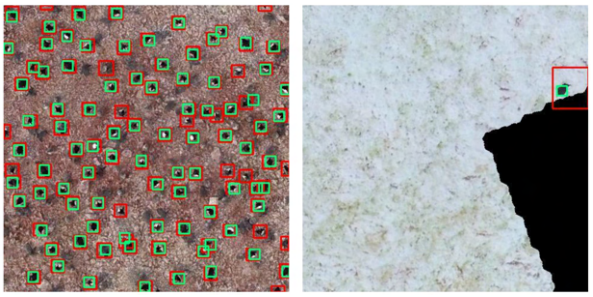


Figure 20: Training parameters grid search for the YOLOv8 model trained on *Penguins-Palmyra*.

B.2 Global model

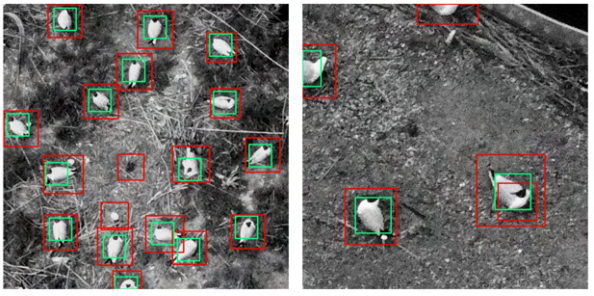


(a) Pfeifer

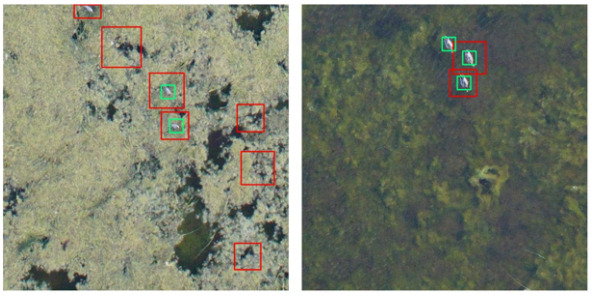


(b) Penguins

(c) Palmyra

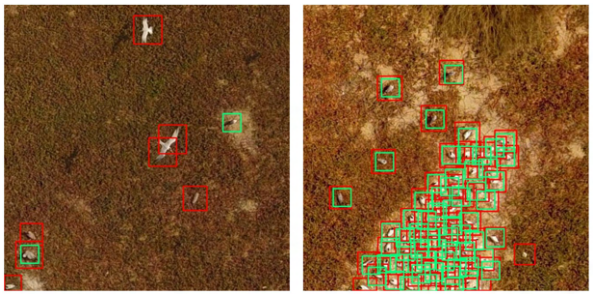


(d) Poland

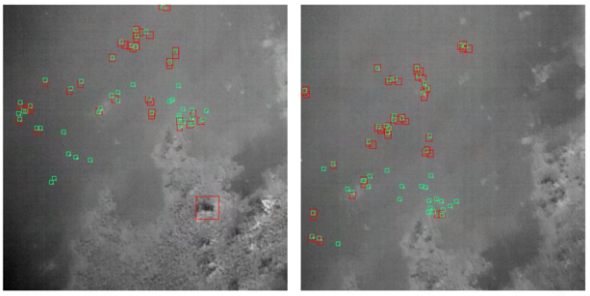


(e) McKellar

(f) Hayes



(g) Terns



(h) Waterfowls

Figure 21: Prediction examples of the global model trained on the global dataset. Groundtruths are in green and predictions are in red.

B.3 Domain adaptation architectures

We report below the grid search and losses analysis graphs of the three adversarial domain adaptation models trained on *Penguins* to *Palmyra*, in the supervised setup.

Figures 22, 25 and 27 show grid search performed to determine gains α of new losses introduced by the domain adaptive subnetworks. The graphs show the average precision values on the source and target domains across multiple loss weights and highlight the respective optimal weight. The same analysis is performed for the second domain shift studied, *Poland* to *McKellar* and the resulting losses gains are gathered in the parameters Table 9.

Figures 23, 26 and 28 introduce training losses evolution to assert the smooth training of tested models. The left graphs show evolution of original detection losses while the right graphs provide losses of the domain adaptive networks. Decreases of the detection losses confirm that the model does learn the detection task. Small increases of the domain adversarial adaptive networks in early epochs verify expected behavior of the subnetwork. Due to the dual objective implemented by the Gradient Reversal Layer, the domain classifiers losses increase as source and target features become more and more indistinguishable, the domain classifier struggle to classify them correctly. The evolution of the domain classifiers accuracies confirm previous behavior observations as visible on Figure 24 for the single domain classifier architecture.

B.3.1 Single Domain Classifier

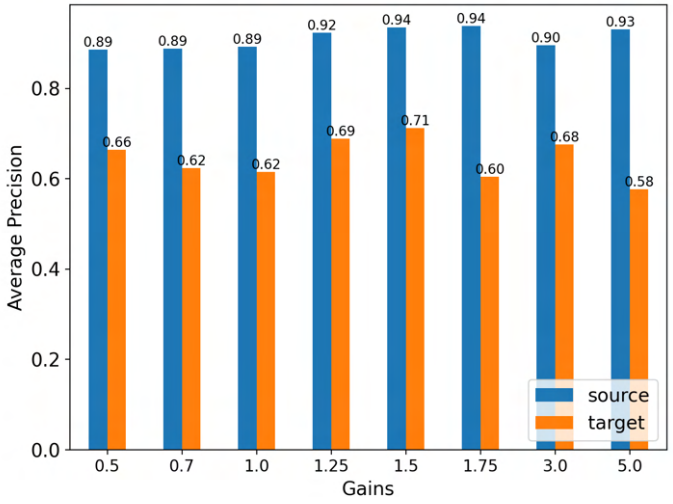
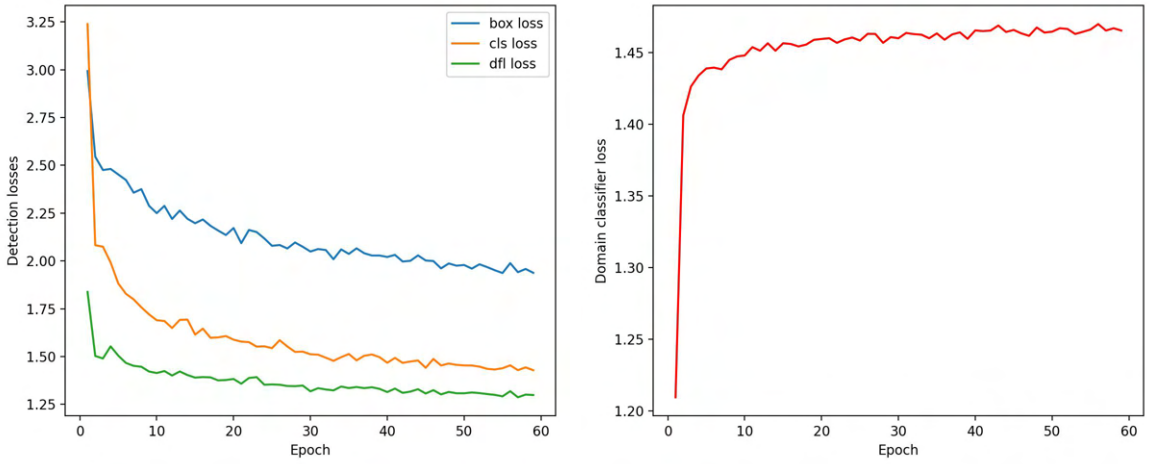


Figure 22: Grid search results for the Single Domain Classifier loss weight. Average precision values on the source and target domains are given for multiple weight values.



(a) YOLOv8 detection losses

(b) Domain classifier network loss

Figure 23: Training losses evolution of the Single Domain Classifier architecture trained on *Penguins* to *Palmyra*.

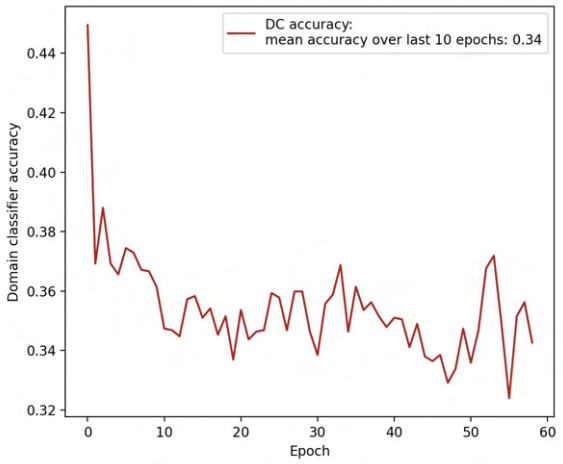


Figure 24: Evolution of the domain classifier accuracy during training of the Single Domain Classifier architecture trained *Penguins* to *Palmyra*.

B.3.2 Multi-Domain Classifiers

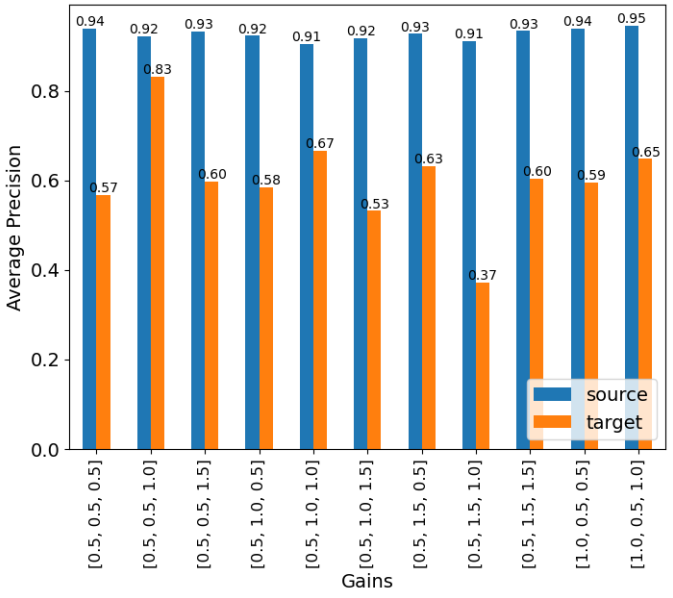
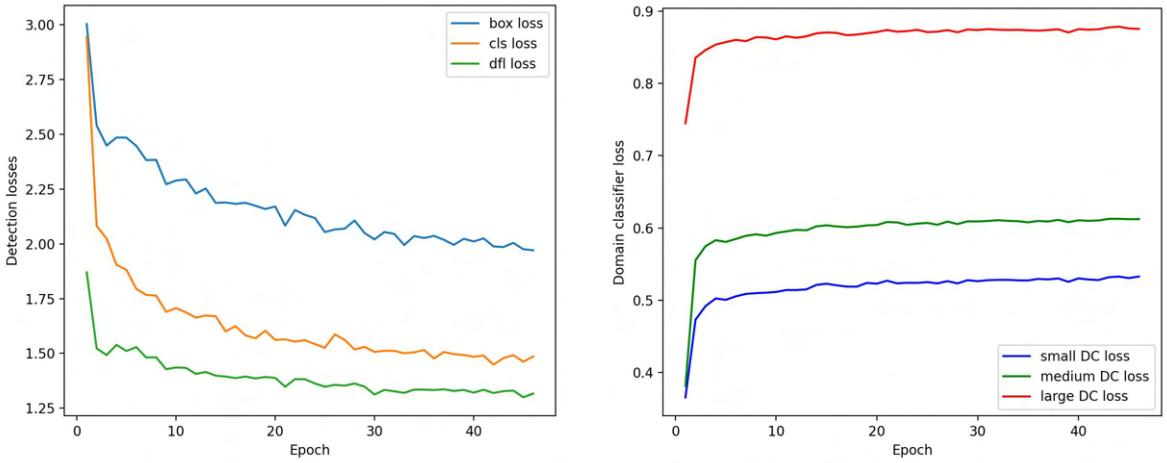


Figure 25: Grid search results for the Multi-Domain Classifier losses weights. Average precision values on the source and target domains are given for multiple weights values.



(a) YOLOv8 detection losses

(b) Domain adaptive network losses

Figure 26: Training losses evolution of the Multi-Domain Classifiers architecture trained on *Penguins* to *Palmyra*.

B.3.3 Multi-Features Domain Classifier

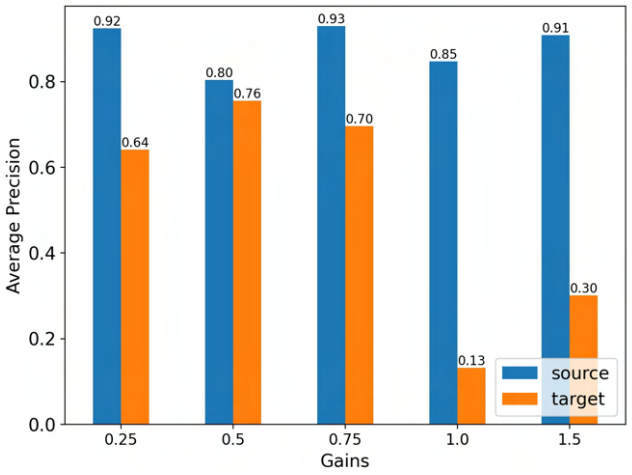
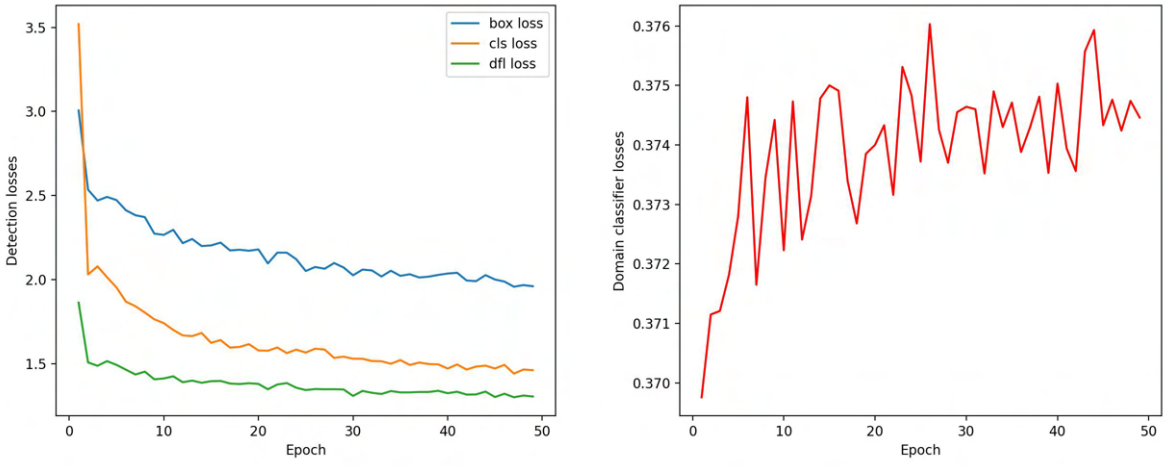


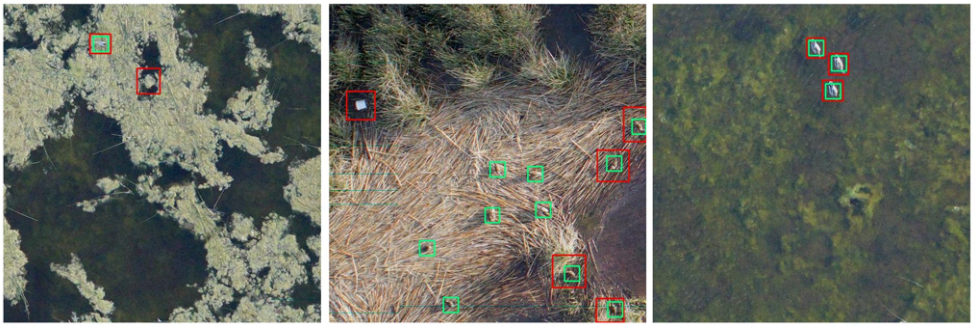
Figure 27: Grid search results for the Multi-Features Domain Classifier loss weight. Average precision values on the source and target domains are given for multiple weight values.



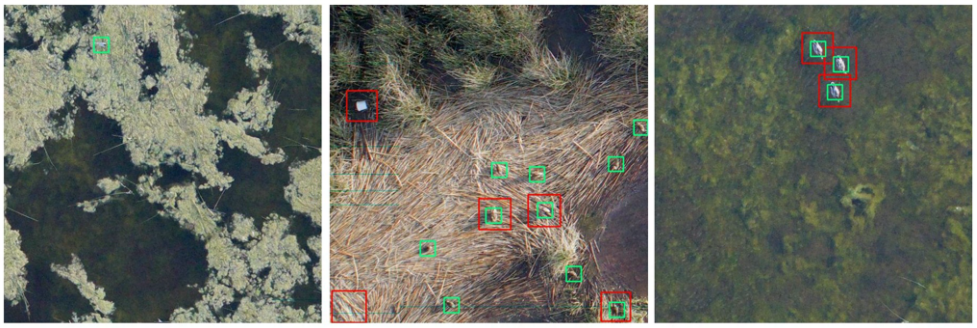
(a) Domain adaptation network loss

(b) Domain adaptive network loss

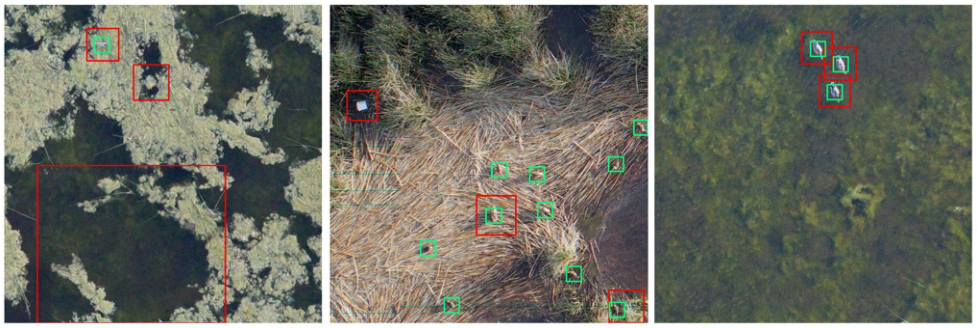
Figure 28: Training losses evolution of the Multi-Features Domain Classifier architecture trained on *Penguins* to *Palmyra*.



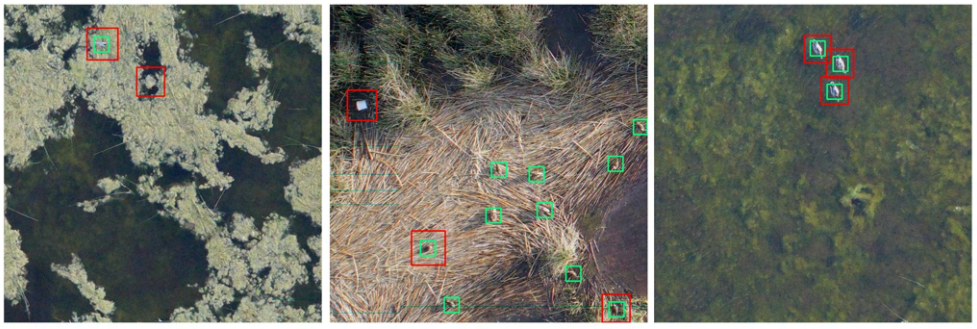
(a) Prediction examples of the YOLOv8 model.



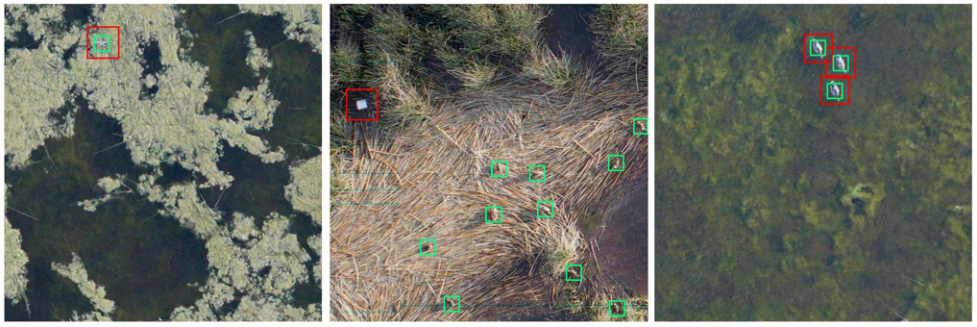
(b) Prediction examples of the Features Alignment using L2-norm architecture.



(c) Prediction examples of the Single Domain Classifier architecture.



(d) Prediction examples of the Multi-Domain Classifiers architecture.



(e) Prediction examples of the Multi-Features Domain Classifier architecture.

Figure 29: Prediction examples of the domain adaptation architectures from *Poland* to *McKellar*. Models are trained in a supervised setup. Groundtruths are in green and predictions are in red.