geomatics

MDPI

Article

Train Fast While Reducing False Positives: Improving Animal Classification Performance Using Convolutional Neural Networks

Mael Moreni [1,2,3,*], Jerome Theau [1,3] and Samuel Foucher [2]

[1] Department of Applied Geomatics, Université de Sherbrooke, Sherbrooke, QC J1K 2R1, Canada; jerome.theau@usherbrooke.ca
[2] Computer Research Institute of Montréal, Montréal, QC H3T 1J4, Canada; samuel.soucher@crim.ca
[3] Quebec Centre for Biodiversity Science (QCBS), Stewart Biology, McGill University, Montréal, QC H3A 1B1, Canada
* Correspondence: mael.moreni@usherbrooke.ca

Abstract: The combination of unmanned aerial vehicles (UAV) with deep learning models has the capacity to replace manned aircrafts for wildlife surveys. However, the scarcity of animals in the wild often leads to highly unbalanced, large datasets for which even a good detection method can return a large amount of false detections. Our objectives in this paper were to design a training method that would reduce training time, decrease the number of false positives and alleviate the fine-tuning effort of an image classifier in a context of animal surveys. We acquired two highly unbalanced datasets of deer images with a UAV and trained a Resnet-18 classifier using hard-negative mining and a series of recent techniques. Our method achieved sub-decimal false positive rates on two test sets (1 false positive per 19,162 and 213,312 negatives respectively), while training on small but relevant fractions of the data. The resulting training times were therefore significantly shorter than they would have been using the whole datasets. This high level of efficiency was achieved with little tuning effort and using simple techniques. We believe this parsimonious approach to dealing with highly unbalanced, large datasets could be particularly useful to projects with either limited resources or extremely large datasets.

Keywords: unmanned aerial vehicles; convolutional neural network; wildlife survey; remote sensing; deep learning; conservation; hard-negative mining

Citation: Moreni, M.; Theau, J.; Foucher, S. Train Fast While Reducing False Positives: Improving Animal Classification Performance Using Convolutional Neural Networks. *Geomatics* **2021**, *1*, 34–49. https://doi.org/10.3390/geomatics1010004

Received: 17 December 2020
Accepted: 12 January 2021
Published: 15 January 2021

1. Introduction

Accurate animal counts are the cornerstone of robust conservation and management plans [1]. For species prone to be in conflict with humans or when populations densities can greatly vary in time and space, they need to be carried out frequently [2]. Many different techniques exist to assess animal populations, from indirect methods, like pellet counts, to direct visual counting [3–5]. Most often, animal censuses are species-specific and require substantial investments in time, money, and effort by wildlife management teams [6]. Whilst some species gather periodically in specific locations, making population assessment easier [5,7], others roam alone or in small groups across vast territories [8,9]. Perhaps the most commonly used technique in open or semi-open environments is direct visual counting. Be it from the ground or from a moving aircraft, it is relatively easy to set up and carry out. However, it is prone to errors due to animal movement, group sizes, poor lines of sights, or variations in the observer's capacities [5,10].

Unmanned aerial vehicles (UAV), commonly known as drones, have recently become more accessible to researchers [11]. They allow easy access to remote areas, are safer and less technically challenging than their manned counterparts, are less stressful for animals and offer the possibility to completely automate flights [2,12,13]. Moreover, the onboard

positioning systems allow the possibility to reproduce earlier flights, making them well-suited for regular assessments of the same areas [2]. When used correctly, they have proven to be able to produce more accurate counts than direct methods [5]. Because they are most often equipped with a digital imagery sensor, they capture the whole scene, and thus allow the counting of big groups of individuals or multiple species. They also offer the possibility to use thermal infrared imagery. This has proven to be effective when detecting animals that wouldn't be visible on RGB images, for instance for animals hidden in tree foliage [14] or at night [15]. All these characteristics make them particularly promising alternatives to standard methods to reduce costs and efforts, and increase the accuracy of wildlife surveys [14,16,17].

So far, the main bottleneck hindering their wide deployment is the difficulty to process the vast amounts of data they generate [2,13,17]. Luckily, automatic object detection has known an important revolution in the last few years, thanks to the use of convolutional neural networks (CNN) [18], making the processing of large amounts of images faster and more accurate than even humans on specific tasks [19,20]. Compared to previous image classification techniques, they are completely data driven, extracting and refining automatically the relevant information to make their decision [21]. Moreover, their performance is known to increase with the amount of data provided [22], making them particularly interesting for tasks that repeatedly collect vast amounts of data, like self-driving cars or in our case, animal census. They have successfully been used to detect various species [8,23–26] and their use in ecology has been on the rise in the past few years [21]. However, obtaining good performances is often the result of tedious trial and error and educated guesses, looking for the right values of the numerous hyperparameters that drive the learning process in a long iterative effort [27].

The sparsity of animals in the wild [8,26] make their detection subject to the false positive paradox, where a detection method with good accuracy might end up giving more false positives (FP) than true positives if the natural frequency of the positive class is extremely low [24,28]. These FP would then have to be processed manually, hindering the performance of automatic detection [28]. Furthermore, this sparsity will naturally lead to collecting many more images of background than of animals. The difference in numbers of samples between classes is called class imbalance and is known to have a negative effect on training deep learning classifiers. [29,30]. This is a common issue in fields such as disease diagnosis or fraud detection, where the events of interest are rare [31,32]. Several methods, such as oversampling, undersampling, class weighting, or thresholding have been studied to tackle this issue [28,33]. While class weighting and oversampling seem to perform better than the others [28,29,33], they have been evaluated by training on whole unbalanced datasets, which can be very time consuming with large datasets.

Most often when using CNN to detect animals, any object that is not the class of interest is labelled as background. As explained by Kellenberger et al. [28], the wider the covered area, the more landscape variety the background class will contain. However, some background objects might not be as common within the background class as others. This intraclass imbalance can in turn be the source of false detections because the network hasn't seen enough of those rare samples [29]. Most of the previously discussed techniques to address class imbalance cannot be used to address this issue because the subclasses are not explicitly labelled within the background class. Furthermore, when using the whole available training data as a training set, the overly represented background objects might end up wasting computing time training on many easy samples and diluting the impact of the hard ones.

Hard-negative mining (HNM), also known as bootstrapping, is the search for negative samples that the network fails to correctly classify [34]. Kellenberger et al. [28] use it to fine tune a network after training it on the whole training set. However, it was originally designed as an iterative process to build the training set by selecting the most relevant samples from the training data. Because only the relevant samples are selected to form the training set, the number of samples it contains is kept to a minimum while maintaining

good levels of performance and short training times. The main downside of this method is that it requires several rounds of training.

In this paper, we present a general method that simultaneously tackles two major hurdles of training neural networks in image classification for wildlife surveys: the high number of FP and the big size of datasets. More specifically, we showcase the effectiveness of HNM to reduce the number of FP while training quickly and efficiently, using a series of recent, simple, and available methods without needing extensive fine-tuning.

2. Materials and Methods

2.1. Data Acquisition

In order to train and test our models, we acquired images of red deer (*Cervus elaphus*), in a deer farm near La Chute, Québec, Canada. This setup allowed us to ensure the presence of around 250 deer in an open, but small, controlled environment. The site is composed of five enclosures of different sizes and vegetation cover (Figure 1b–d).

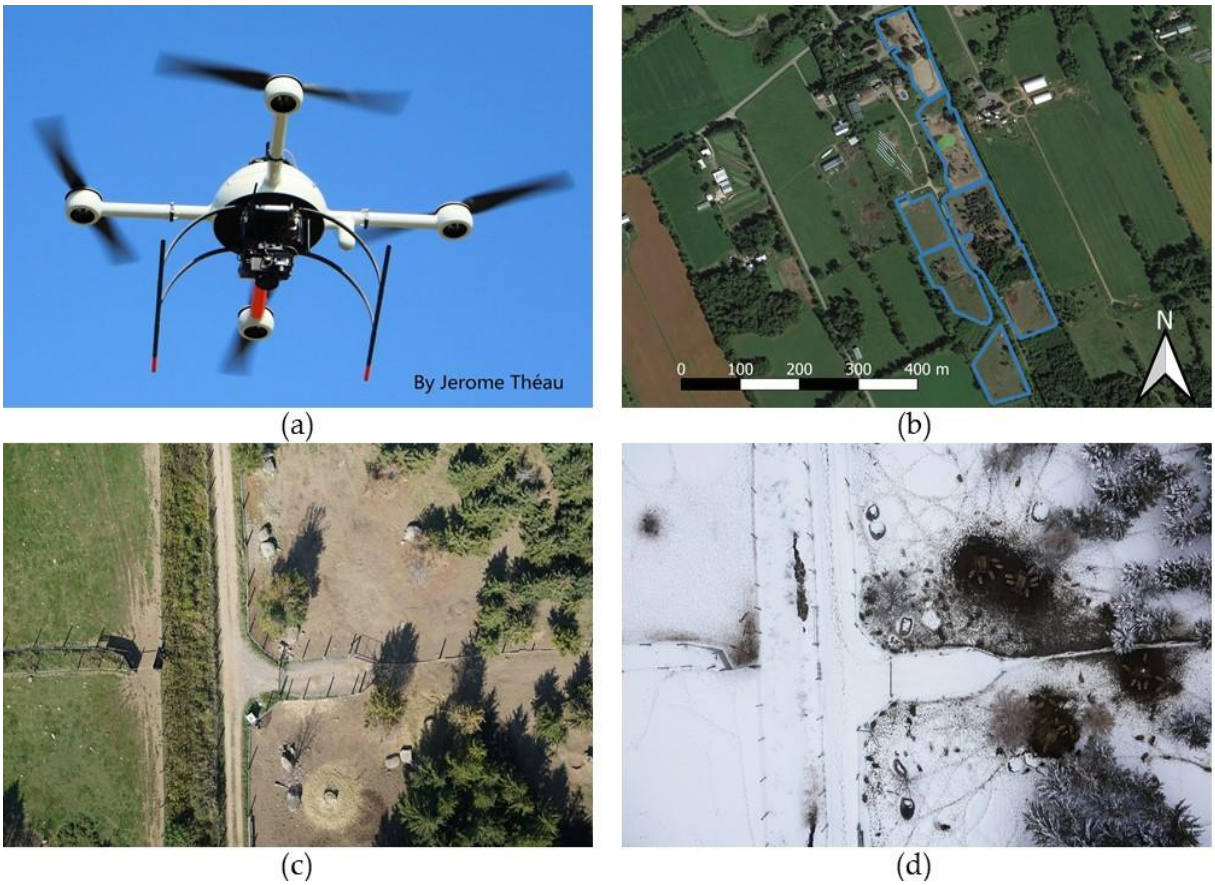

Figure 1. Pictures of the unmanned aerial vehicle (UAV) used for the data acquisition (**a**); outline of the enclosures (**b**); examples of images of the same area in the summer and winter (**c**,**d**).

We used an electric multirotor UAV from Microdrones (Berlin, Germany), the md4-1000 (Figure 1a), equipped with a Sony RXI RII camera and a 35 mm lens. The camera takes RGB images of 7852 × 5304 pixels.

We flew over the site twice, in the summer on 25 August 2017 and in the winter on 3 March 2018 in order to get two sets of images under very different environmental conditions, to ensure we had a variety of backgrounds. In the summer, some of the enclosures were very dry, covered with bare soil or dry grass. Only one enclosure had green grass. In the winter, the deer were grouped in three enclosures. Most of the ground was covered in snow but the rising temperatures were leaving patches of bare soil where the deer gathered (Figure 1c,d). From the UAVs perspective, the deer were either standing

up or lying down, exposing their flanks. While this species can grow to be more than 2 m long, most of the herd were young individuals measuring around 1.6 m long or less.

Flights were done at 40 and 80 m above ground level in order to diversify the dataset. In the summer acquisition, we flew over each deer enclosure at least once, at both altitudes. Takeoff and landing were performed manually, away from the deer to avoid causing stress. The rest of the flights were carried out automatically along linear transects covering the whole enclosure. Images were taken with 80% frontal and lateral overlap. In the winter, the colder temperatures reduced battery life. For safety reasons, we flew over all the enclosures every flight.

2.2. Image Pre-Processing

For each acquisition, the images were grouped by flight and were split between positive images containing deer, and negative images not containing any. The positive images were manually annotated in ArcMap 10.6 from Esri, where each deer was marked by a point in a vector layer. Individual images of deer were extracted from square windows (of 350 × 350 pixels and 175 × 175 pixels at 40 and 80 m respectively, or about 3 m^2, with ground sample distances (GSD) of 5.2 mm and 10.3 mm respectively) centered on each detection. These individual images were then sorted to only contain whole and unobstructed images of deer. Their negative counterparts were automatically generated by cropping the large negative images along a sliding window of the same size as the ones used for the individual images of deer. A random sample of the resulting small negative images was selected for each flight to match the number of positive images, thus creating a balanced binary classification dataset (hereafter cited as initial datasets).

The training, validation, and test sets were made of images from separate flights to avoid testing the network on images very similar to the ones it had already been trained on (due to the high overlap).

Unbalanced datasets were also created for the training and validation sets, containing all the available negative images. These datasets acted as the hard-negative 'mines' on which we ran our trained model to retrieve hard samples. We will refer to these datasets as training and validation "pools" to water down the mining references in the paper. The resulting datasets and their imbalance factor (the number of negative samples per positive sample) are summarized in Table 1.

Table 1. Sizes (number of images) of the initial training and validation sets, pools, and test sets and their imbalance factors for the winter and summer acquisitions.

	Initial Training Set	Initial Validation Set	Training Pool	Validation Pool	Test Set
			Summer		
Deer	3307	911	3307	911	1209
Negative	3307	911	627,131	85,923	325,756
Imbalance factor	1	1			
			Winter		
Deer	3434	1203	3434	1203	1518
Negative	3434	1203	424,647	77,100	426,625
Imbalance factor	1	1	193	64	281

2.3. Proposed Approach

The training and testing were carried out using Pytorch 1.1.0 [35] on three graphics processing units (GPU)(NVIDIA GeForce GTX 1080 Ti) with a Resnet-18 [36] pre-trained on Imagnet, from the Pytorch model zoo. Using the smallest network of the Resnet family gave good results while reducing training time compared to deeper architectures. For this binary classification task, the capacity of the network was enough to obtain good results in a reasonable amount of time. Because the images on ImageNet are likely to be very

different from aerial images, we decided to fine-tune the whole network instead of specific layers.

In our training, we used the optimizer called Ranger, implemented by Less Wright (https://github.com/lessw2020/Ranger-Deep-Learning-Optimizer), that combines the recent techniques of Rectified Adam [37] and LookAhead [38], with a flat learning rate schedule and a batch size of 1100.

As we planned to train on data that the network had already seen through the HNM process, we expected overfitting (the loss of generalization performance after training too long on the same data) to be more present than if we were training from scratch every time. To address this issue, we used data augmentation at every epoch on the training data, composed of random flips, rotations, rescaling, and brightness changes as well as early stopping [39] with a patience of 20 epochs. The small size of our validation set allowed us to assess the validation performance after each epoch.

To select the learning rate, we used an implementation of the learning rate range test (LRRT) [40] (https://github.com/davidtvs/pytorch-lr-finder). This simple test allowed us to pick the learning rate without a grid search, therefore saving a lot of fine-tuning time.

2.3.1. Metric

The metric is the score by which the performance of the model is evaluated, either during training on the validation dataset or after training on the test set. The most common metrics for binary classification tasks are the accuracy and the F1 score [41]. However, these metrics can be misleading and generate overoptimistic results when applied to unbalanced datasets [41]. Instead, we chose to use the Matthews correlation coefficient (MCC), which requires the classifier to perform well on both negative and positive samples to get a good score, regardless of their ratio [41].

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}} \quad (1)$$

TP = True Positives, TN = True Negatives, FP = False Positives, FN = False Negatives.

Because we expect the training data to become more unbalanced as the wildlife survey progresses, the MCC assures us that the score given will be consistent and unbiased by the imbalance throughout the process.

2.3.2. Workflow

Proposed Hard-Negative Mining (HNM)

For both acquisitions, we started our training procedure with balanced training and validation sets (Table 1) on which we ran our model with 10 different seeds. We called this round of training "round 0" as no hard sample had been found yet. We used the validation MCC score to select the best model and ran it in inference mode on the training and validation pools to select the hard samples. Only new images not already present in the validation set were added, as opposed to the training set for which all images were added. We ran this process until the number of hard negatives reached our acceptability threshold of 10 FP on the validation pool. To make sure that the gains transferred well to the test set, we ran the best model of each training round on the test set, but all the decisions regarding the training were made based on the validation set. We modified the inference process to also output the class activation maps [42] of the incorrectly classified images, in order to visualize the areas the network was basing its decisions on. The whole process is summarized in Figure 2.

Training Times

For a given hardware setup, the training time directly depends on the number of images present in the training and validation sets. To compare the training time of our method to the training on the full dataset, we ran a training run on them for both acquisitions with the same techniques and optimizer used in HNM. For each acquisition, we

performed a LRRT to pick a learning rate value, then trained the same pretrained network used in round 0 of HNM with an early stopping patience of five epochs.

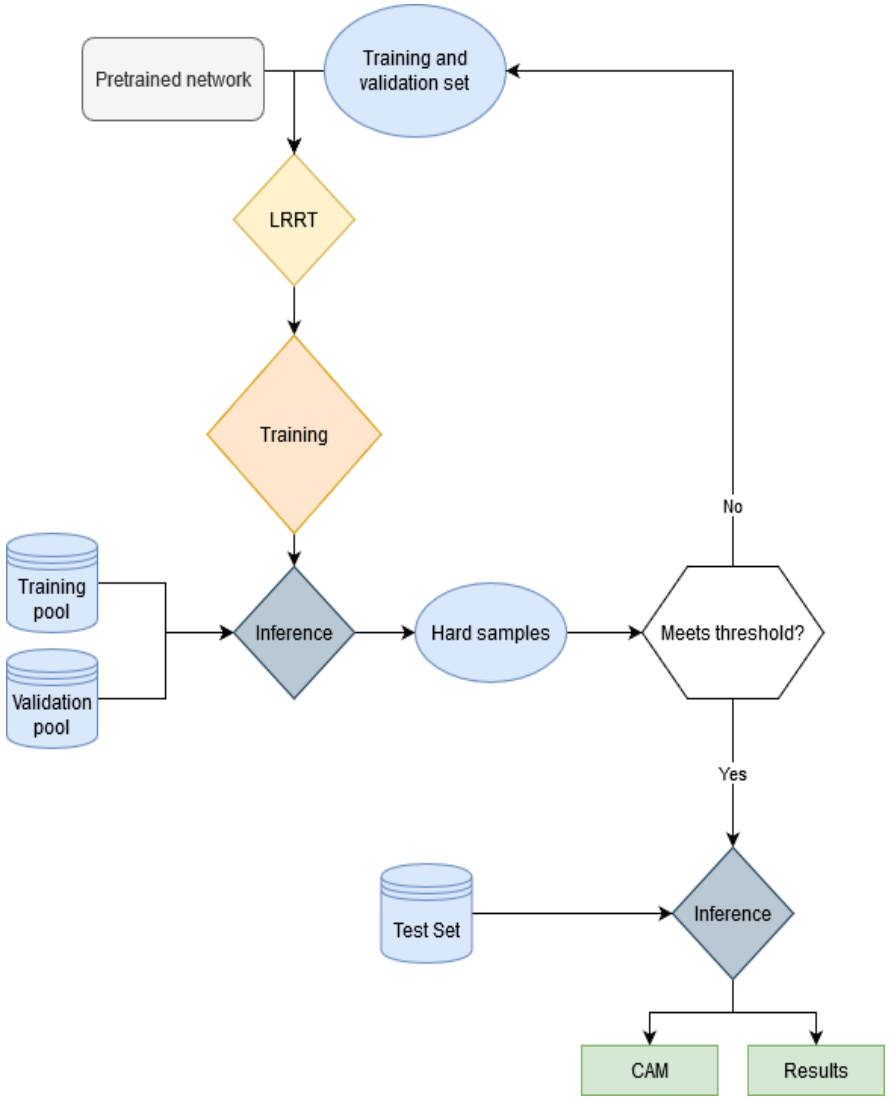

Figure 2. Summary of the hard-negative mining procedure.

3. Results

3.1. Hard-Negative Mining

It took only one round of HNM for both acquisitions to reach our acceptability threshold. The details of the training and validation sets for both acquisitions can be found in Table 2.

The inference on the training and validation pools after round 0 of HNM added 2837 and 854 negative samples to the summer training and validation set, bringing their imbalance factors from 1 to 1.86 and 1.94 respectively. Much fewer hard samples were found for the winter acquisition, with 53 added to the training set and 51 to the validation set. The imbalance factors went from 1 to 1.02 and 1.04 respectively. The proportions of hard samples in the full dataset were larger for the summer acquisition, with 0.45% and 0.99% for the training and validation pools respectively, against 0.01% and 0.06% for the winter dataset.

For round 1 of training, the summer training and validation sets were made of 0.98 and 2.06% of the training and validation pools respectively. For the winter acquisition, they represented 1.44% and 1.63% of the training and validation pools.

Table 2. Number of images and characteristics of training and validation sets and pools for both summer and winter acquisitions for the two rounds of training.

		Training Set	% of Training Pool	Training Pool	Validation Set	% of Validation Pool	Validation Pool
				Summer			
Round 0	Deer	3307	100	3434	911	100	911
	Negative	3307	0.53	627,131	911	1.06	85,923
Round 1	Deer	3307	100	3307	911	100	911
	Negative	6144	0.98	627,131	1765	2.06	85,923
	Imbalance factor	2		190	2		94
				Winter			
Round 0	Deer	3434	100	3434	1203	100	1203
	Negative	3434	0.81	426,747	1203	1.56	77,100
Round 1	Deer	3434	100	3434	1203	100	1203
	Negative	3487	1.44	426,747	1254	1.63	77,100
	Imbalance factor	1		124	1		64

Adding these hard samples had some notable impacts on the performance on the validation pool sets for both acquisitions after round 1 of training (Tables 3 and 4). The numbers of FP went from 854 to 4 and 51 to 5 for the summer and winter sets respectively, thereby reducing the FP rates by 99.5% and 90.2% respectively. The numbers of FN increased for both acquisitions between the two rounds of training, going from 0 to 9 for the summer acquisition and from 0 to 1 for the winter acquisition. However, round 1 of training still resulted in an overall gain in performance, as the MCC scores went from 0.715 to 0.993 for the summer acquisition and from 0.979 to 0.998 for the winter acquisition. The summer acquisition showed the highest gain from the HNM process, as the MCC on the validation pool improved by 28% between the two rounds, compared to the 1.9% increase in the winter validation pool. The FP rate on the validation pool at the end of round 1 of training was 1 FP for 21,276 negative images for the summer set and 1 FP for 15,385 negative images for the winter set. This transferred well to both test sets, which went from 1/185 and 1/13,761 to 1/19,162 and 1/213,312 for the summer and winter sets respectively (Tables 3 and 4).

Table 3. Confusion matrix of the inference on the summer validation pool and test set after both rounds of training and their Matthews correlation coefficient (MCC) scores.

		Round 0			Round 1		
		Predicted Label			Predicted Label		
		Deer	Negative	MCC	Deer	Negative	MCC
Validation /True label	Deer	911	0	0.7149	902	9	0.9928
	Negative	854	85,069		4	85,919	
Test /True label	Deer	1157	52	0.6142	1192	17	0.9859
	Negative	1760	323,996		17	325,739	

Table 4. Confusion matrix of the inference on the winter validation pool and test set after both rounds of training and their Matthews correlation coefficient (MCC) scores.

		Round 0			Round 1		
		Predicted Label			Predicted Label		
		Deer	Negative	MCC	Deer	Negative	MCC
Validation /True label	Deer	1203	0	0.9791	1202	1	0.9975
	Negative	51	77,049		5	77,095	
Test /True label	Deer	1516	2	0.9883	1502	16	0.9940
	Negative	34	426,591		2	426,623	

3.2. Class Activation Maps

The class activation maps of most of the false negatives show that the decision to classify these images as negatives is based on the background surrounding the deer, avoiding the deer itself (Figure 3a).

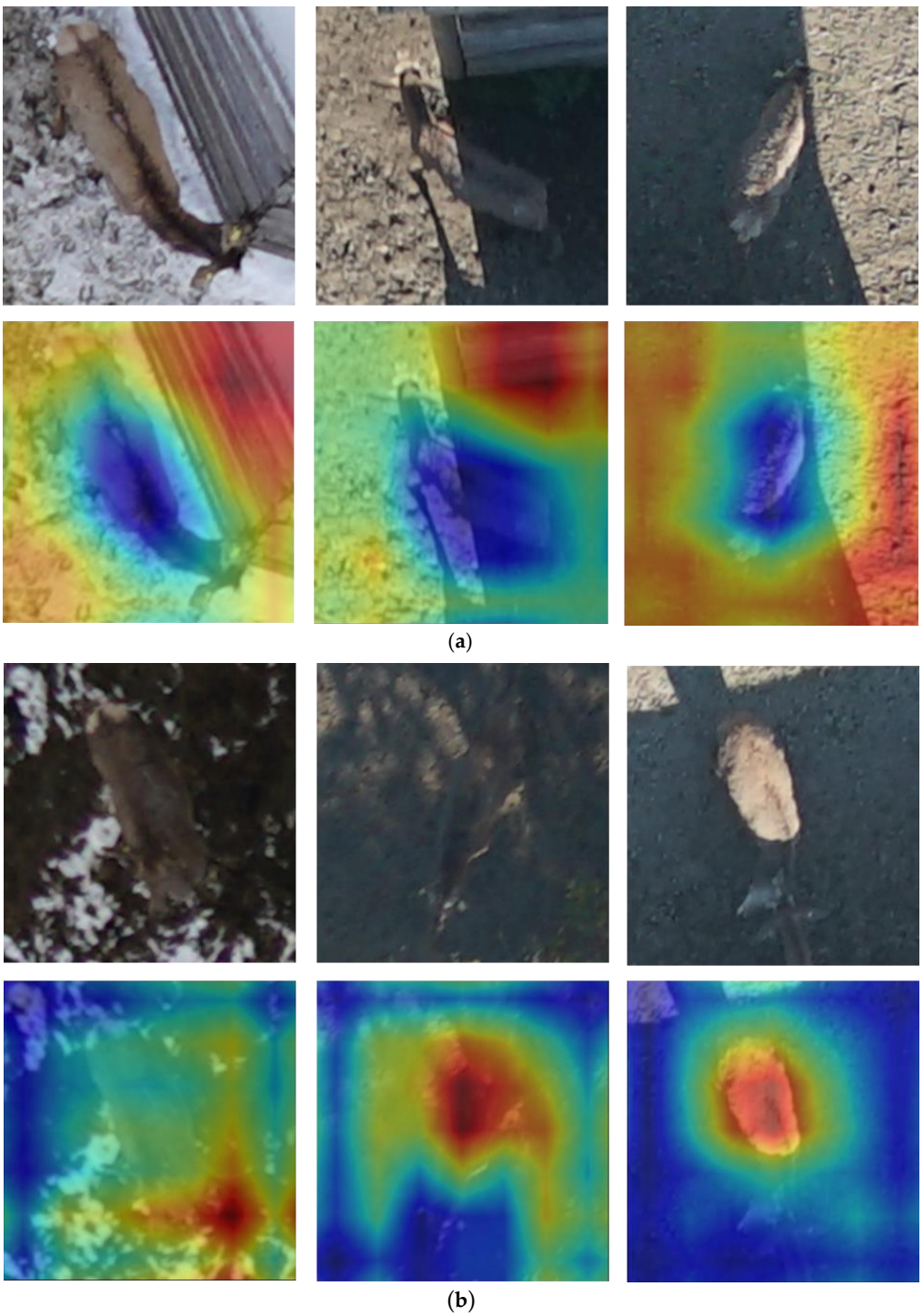

Figure 3. Examples of false negatives and their respective class activation map. The warm-colored areas are the ones that led to the decision to classify the image as a negative, while the cold colors contributed the least. Image (**a**) shows images in which the network perceived a difference between the deer and the background; (**b**) shows the cases in which it didn't detect the deer or mistook it for background.

The network failed to distinguish between the background and the deer on only two images on the summer test set and one on the winter test set (Figure 3b).

3.3. Training Times

The difference in training time between the two rounds of training of HNM and training on the full dataset for both acquisitions is summarized in Tables 5 and 6.

Table 5. Comparison of training times between the two rounds of hard-negative mining (HNM) and training on the full unbalanced dataset for the summer and winter acquisitions.

	Total Number of Epochs	Total Training Time (min)	Average Training Time Per Model (min)
		Summer	
HNM	1150	1068	106.8
Full dataset	10	932	932
		Winter	
HNM	1135	872	87.2
Full dataset	14	295	295

Table 6. Performance on validation pool and on test sets for the best model of the hard-negative mining (HNM) process and the model trained on the full unbalanced dataset for both acquisitions.

	Validation Pool MCC	Test Set MCC
	Summer	
Best HNM model	0.993	0.986
Full dataset	0.877	0.923
	Winter	
Best HNM model	0.998	0.994
Full dataset	0.996	0.990

One training on the full summer dataset took 932 min (15 h and 32 min). This is 136 min less than it took to train the 20 models with HNM. The training on the full winter dataset was much faster than for the summer dataset (295 min or 4 h and 55 min), a little more than a third of the time it took to train the 20 models of the HNM on the same acquisition. For both acquisitions, the final models of the HNM process outperformed the models trained on the full datasets on the validation and test MCC scores, by 13.19% and 6.79% on the summer validation and test MCC respectively and by 0.17% and 0.36% for the winter acquisition. On average, it took 106.8 and 87.2 min to train a single model through the HNM method for the summer and winter acquisitions respectively.

4. Discussion

Our goal in this paper was to reduce training time and the number of FN when training on highly unbalanced data, with minimal fine-tuning of hyperparameters. Our method managed to achieve better performance than the same model on the whole dataset, in a fraction of the training time and with very low FP rates (around 1 FP per 6 hectares on the summer test set and 1 per 60 hectares on the winter test set).

4.1. Training the Models

Early stopping allowed us to simultaneously avoid overfitting, save the version of the network with the best generalization performance, and limit training time. However, shortening the training also limits the capacity of an optimizer such as RAdam to reach its best performance when using a suboptimal learning rate. The LRRT offers an interesting synergy with early stopping in this regard, as it ensures that the learning rate is picked

within the range containing the optimal value. Therefore, we can expect good performance from the beginning of training and an end result not too far from what it would have been with the optimal value. While training parameters are generally given in the literature, little information is provided regarding the process to pick them [8,24,43,44]. The LRRT used in this article offers an interesting tool to standardize the search for good learning rate values at little cost. Moreover, it can also be used to find good values for other optimizer-related parameters, such as weight decay or momentum [27].

Slightly worse hyperparameter values than the ones chosen would likely have increased the number of hard samples and the time for the network to converge, but the performance between HNM rounds would likely have improved due to the addition of new, relevant data. In our eyes, the method presented here offers a good trade-off between shortening the training time and good generalization performance. An alternative approach to improve the performance on the final round of training (round 1 in this case) would be to spend more time fine-tuning the hyperparameters instead of using the same exact training method as in the previous rounds. However, in a case like ours where the performance on the validation set is already very good, we expect diminishing returns on the time invested in the fine-tuning.

Apart from the impact high levels of imbalance can have on generalization performance, training on large datasets requires a lot of computing time. Moreover, the hyperparameter tuning required to use any method that tackles imbalance on a full dataset, also takes significantly longer on a large dataset. While HNM doesn't completely remove the imbalance, it greatly reduces its magnitude. The other techniques to mitigate the impact of imbalance mentioned earlier could still be used in conjunction with HNM, but the fine-tuning of their parameters would require far less time due to the smaller size of the training and validation sets.

We noticed high levels of variability between runs using different random initializations for a given set of hyperparameters, despite what can be read in [43], and therefore encourage practitioners to try several runs before settling on a final model (see Appendix A for more details).

4.2. Hard-Negative Mining

Whilst the FP rate was better on the winter acquisition than on the summer one, the HNM impact on the classification performance between the two rounds of training was much stronger on the summer acquisition (improvement of a factor 100 on the summer test set against an improvement of a factor 14 on the winter test set). We believe the reason for this to be the higher variety of objects present in the negative class of the summer dataset compared to its winter counterpart. In the winter, snow covered the majority of the objects present in the images, thereby reducing the intraclass imbalance of the negative class while increasing the contrast between the deer and the background. With more objects to confuse the network in the summer acquisition, the first inference on the training and validation pools returned significantly more FP than for the winter dataset. Most of these FP were similar, with a vast majority of them being of rocks, tree trunks, or shadows and happened to be almost absent of the initial training set. In this regard, the HNM ended up being a way of oversampling confusing objects within the negative class of the training set.

While applying the network to new areas that may offer different background diversity than previously encountered and thus may decrease its performance [28], the HNM process can retrieve only the informative examples needed to fine-tune the network. This would allow the training process to scale well in time as more data is acquired, without needing to retrain the network from scratch.

Unsurprisingly, the training of a model through HNM was significantly faster than a simple training on the full dataset and achieved better results. This highlights the negative impact a high number of easy samples can have on performance when nothing is done to mitigate the imbalance.

We believe this approach could be very beneficial to studies that use CNN to perform image classification on imbalanced datasets applied to different species, either on camera-trap images [43,44] or on UAV imagery [24]. The latter is a particularly good example as it has annotation and classification methodologies very similar to ours but with a much higher proportion of FP (1 FP for 530 negative images and an MCC score of 0.3526). The vast majority of their negative class is made of ocean, with very little intraclass diversity and therefore few objects that could confuse the network. Most of the negative images are likely to be easy samples that negatively impact the network's performance and could be removed from the training set through HNM. However, the difference in overall performance between our study and theirs doesn't only come from our HNM method. As explained in their article, other factors such as network depth (4 layers against 18 in ours), the use of a non-pretrained network, or the fact that they favor the recall against the precision may also have a significant impact on their network's performance. We could expect similar results to ours for images of similar GSD, of animals of comparable sizes to the red deer and in a similar environment, such as white-tailed deer (Odocoileus *virginianus*), caribou (*Rangifer tarandus*), or black bear (*Ursus americanus*).

When facing high levels of imbalance (75% of their images are negative), Norouzzadeh et al. [43] opted for a two-stage pipeline, first separating empty and full (containing animals) images then classifying the species present in the full images. To that end, they first randomly selected negative images to match the number of positive images, as we do to start our round 0. However, they then carried out their training without using the rest of the negative images, amounting to half of their available data. A single round of mining on the negative data might have brought new informative samples, improving the ability of the network to distinguish between empty and full images, without causing too much imbalance. Perhaps this alone might have helped improve the performance of their one stage pipeline and reduced the need for a two-stage approach.

4.3. Perspectives on Future Work

The nature of image classification as it is performed here can lead to mistakes when the network is more confident in the background area around the deer than the deer itself. When looking at the class activation maps of most of the FN (Figure 3), both deer and background are properly distinguished by the network, but the prediction is not the one we expect. Preliminary testing on these images have shown that cropping a significant portion of the background area around the deer led the network to classify it as deer. We interpret this as the consequence of forcing the network to give only one, non-spatially-specific label to the images containing both classes, based on the one that gives the highest score. A promising avenue to improve on this is to use the same network to perform coarse semantic segmentation by transforming it into a fully convolutional network (FCN). This method outputs a raster per class, highlighting the areas in the image where the class is detected (Figure 4). Similar ideas in Kellenberger et al. [28] and Bowler et al. [26] achieved good levels of performance. We believe that this technique could be used as a detection method but additional work to fully automate the detection from the coarse segmentation map is needed to assess its effectiveness on full-size images.

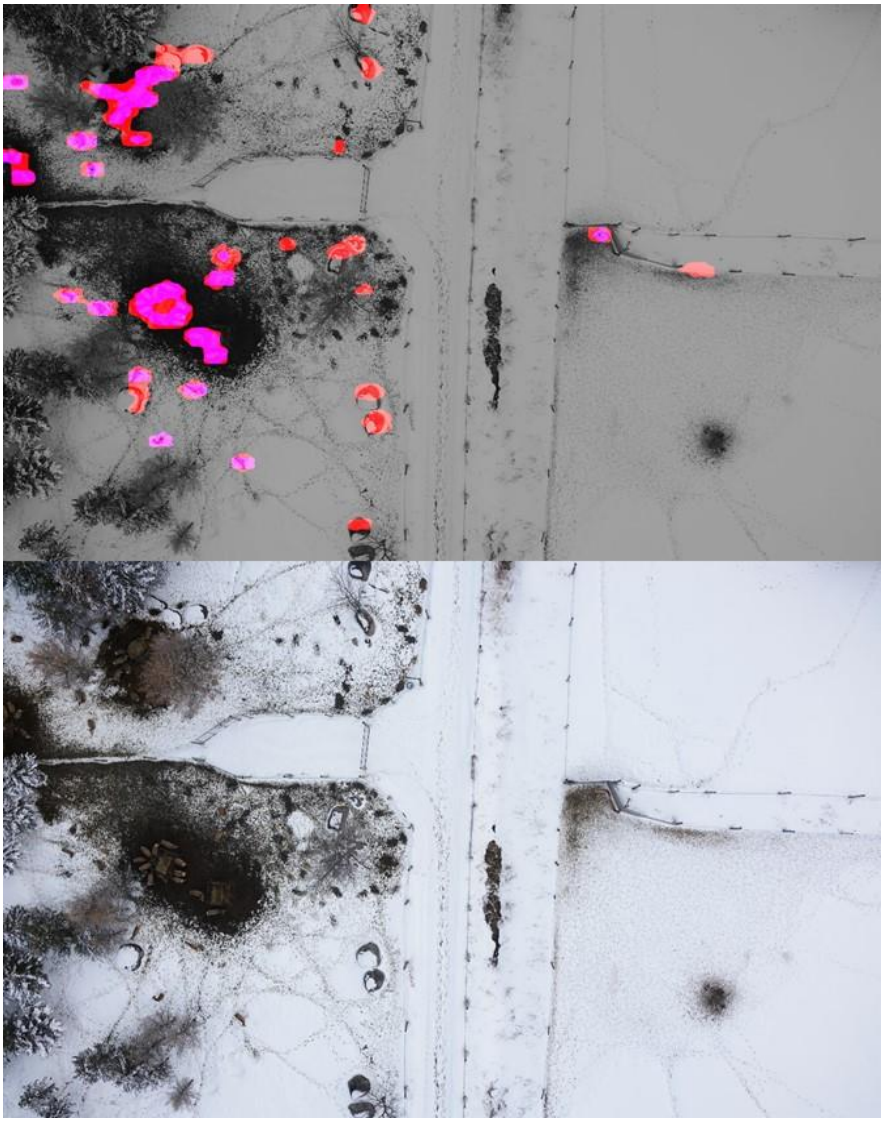

Figure 4. Example of fully convolutional network (FCN) output (bottom) from a full-size image (top). Highlighted in red are the areas detected as deer in round 0, in blue the areas detected as deer in round 1, and in purple the detected areas in both rounds.

Author Contributions: Conceptualization, M.M., S.F., and J.T.; methodology, M.M., J.T., and S.F.; software, M.M.; formal analysis, M.M. and S.F.; resources, S.F.; data curation, M.M.; writing—original draft preparation, M.M.; writing—review and editing, S.F. and J.T.; visualization, M.M.; supervision, S.F. and J.T. All authors have read and agreed to the published version of the manuscript.

Funding: This project was financed in part by the Ministère de l'Économie et de l'Innovation (MEI) of the province of Québec. We are also grateful for the support and funding provided by MITACS grant number IT07901, the Quebec Center for Geomatics, and Microdrones that made this research possible.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to proprietary rights and could only be used for non-commercial purposes.

Acknowledgments: We would like to thank the farm owner, André Viau, who was kind enough to let us access his property.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results. Microdrones collected the images.

Appendix A

Randomness has an important impact on training, from the initialization of the network to the order with which the samples are picked. Setting this randomness from the start is necessary to ensure that training runs are reproducible. This is easily done in Pytorch by setting the seed at the beginning of training. The seed is a number from which python generates pseudo-random numbers. From each seed, the number generated at the Nth call of a function using that given seed will always be the same, thus ensuring run reproducibility. However, because of the randomness it generates throughout the training, it is impossible to predict which seed will yield the best results. It is therefore necessary to use different seeds for each training run and keep the one that performs the best on the validation set.

We randomly picked 10 numbers between 1 and 10,000 to use as seeds. Every round of training was carried out 10 times, using a different seed from our list of seeds.

We noticed that by selecting the best models based on their validation MCC, the performance on the number of false positives on the validation and test pools were both times under the average for the ten different seeds (Tables A1 and A2). However, for both acquisitions, a wrong seed could more than double the number of false positives compared to the lowest value.

Table A1. Details of the performance for each seed used at round 0 of training on the summer dataset, highlighted in blue is the model selected for inference to create the round 1 dataset.

Seed	MCC Val	ValPool MCC	ValPool FP	MCC Test	Test FP
123	0.9891	0.6241	1390	0.6164	1903
2167	0.9771	0.5311	2217	0.4603	4303
1545	0.9803	0.5829	1702	0.5549	2620
8297	0.9879	0.6568	1158	0.6060	1992
1487	0.9858	0.5724	1806	0.5546	2576
7725	0.9847	0.5914	1630	0.5459	2746
8780	0.9913	0.6096	1498	0.5660	2504
5027	0.9836	0.5857	1683	0.4648	4249
970	0.9858	0.6285	1359	0.5328	2945
960	0.9923	0.7149	854	0.6142	1760
Average	0.9858	0.6098	1529.7	0.5516	2759.8
SD	0.0047	0.0506	373.6252	0.0553	886.6702

Table A2. Details of the performance for each seed used at round 0 of training on the winter dataset, highlighted in blue is the model selected for inference to create the round 1 dataset.

Seed	MCC Val	ValPool MCC	ValPool FP	MCC Test	Test FP
123	1.0000	0.9791	51	0.9883	34
2167	0.9983	0.9791	51	0.9879	31
1545	0.9992	0.9752	61	0.9879	34
8297	0.9983	0.9772	56	0.9925	20
1487	0.9983	0.9729	67	0.9867	38
7725	0.9992	0.9791	51	0.9886	31
8780	0.9992	0.9748	62	0.9854	42
5027	0.9992	0.9868	32	0.9931	20
970	0.9983	0.9633	92	0.9807	59
960	0.9992	0.9701	74	0.9863	36
Average	0.9989	0.9758	59.7	0.9877	34.5
SD	0.0006	0.0063	16.042	0.0006	11.138

The seed seems to have a non-negligible effect on the network's performance. Our understanding is that a good initialization, through a favorable seed, may position the network in a spot allowing it to reach a smaller loss faster than with another, less favorable, seed. Although this may not surprise the most experienced practitioners, we thought it could be beneficial to newcomers in the field.

Running a model with multiple seeds is a way to assess the model's average performance. In a real case scenario however, our end goal wouldn't be to have a good assessment of the average performance of a tuned model but to pick the best performing model we could get in a reasonable time. In the case of an iterative process such as ours, selecting the best performing networks early is likely to save even more computing time down the line. Therefore, we would recommend trying out at least three seeds after the hyperparameter selection and keeping the one that performs the best on the validation set. In our case, the small size of our training sets allowed us to test 10 different seeds without committing too much computing time. The number of seeds to try is left at the discretion of the practitioner as it depends on the availability of resources and time constraints. However, we believe that the possible gain in performance compared to the little effort needed to try different seeds make it worth testing.

References

1. Jachmann, H. *Estimating Abundance of African Wildlife: An Aid to Adaptive Management*, 1st ed.; Springer Science & Business Media: Berlin, Germany, 2012; ISBN 978-1-4615-1381-0.
2. Linchant, J.; Lisein, J.; Semeki, J.; Lejeune, P.; Vermeulen, C. Are Unmanned Aircraft Systems (UASs) the Future of Wildlife Monitoring? A Review of Accomplishments and Challenges: A Review of UASs in Wildlife Monitoring. *Mamm. Rev.* **2015**, *45*, 239–252. [CrossRef]
3. Goode, M.J.; Beaver, J.T.; Muller, L.I.; Clark, J.D.; van Manen, F.T.; Harper, C.A.; Basinger, P.S. Capture—Recapture of White-Tailed Deer Using DNA from Fecal Pellet Groups. *Wildl. Biol.* **2014**, *20*, 270–278. [CrossRef]
4. Corlatti, L.; Gugiatti, A.; Pedrotti, L. Spring Spotlight Counts Provide Reliable Indices to Track Changes in Population Size of Mountain-Dwelling Red Deer Cervus Elaphus. *Wildl. Biol.* **2016**, *22*, 268–276. [CrossRef]
5. Hodgson, J.C.; Baylis, S.M.; Mott, R.; Herrod, A.; Clarke, R.H. Precision Wildlife Monitoring Using Unmanned Aerial Vehicles. *Sci. Rep.* **2016**, *6*, 22574. [CrossRef] [PubMed]
6. Witmer, G.W. Wildlife Population Monitoring: Some Practical Considerations. *Wildl. Res.* **2005**, *32*, 259. [CrossRef]
7. Lhoest, S.; Linchant, J.; Quevauvillers, S.; Vermeulen, C.; Lejeune, P. How Many Hippos (Homhip)—Algorithm for Automatic Counts of Animals with Infrared Thermal Imagery from UAV. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2015**, *XL-3/W3*, 355–362. [CrossRef]

8. Rey, N.; Volpi, M.; Joost, S.; Tuia, D. Detecting Animals in African Savanna with UAVs and the Crowds. *Remote Sens. Environ.* **2017**, *200*, 341–351. [CrossRef]
9. Vermeulen, C.; Lejeune, P.; Lisein, J.; Sawadogo, P.; Bouché, P. Unmanned Aerial Survey of Elephants. *PLoS ONE* **2013**, *8*, e54700. [CrossRef]
10. Ransom, J.I. Detection Probability in Aerial Surveys of Feral Horses. *J. Wildl. Manag.* **2012**, *76*, 299. [CrossRef]
11. Burke, C.; Rashman, M.; Wich, S.; Symons, A.; Theron, C.; Longmore, S. Optimizing Observing Strategies for Monitoring Animals Using Drone-Mounted Thermal Infrared Cameras. *Int. J. Remote Sens.* **2019**, *40*, 439–467. [CrossRef]
12. Wang, D.; Shao, Q.; Yue, H. Surveying Wild Animals from Satellites, Manned Aircraft and Unmanned Aerial Systems (UASs): A Review. *Remote Sens.* **2019**, *11*, 1308. [CrossRef]
13. Van Gemert, J.C.; Verschoor, C.R.; Mettes, P.; Epema, K.; Koh, L.P.; Wich, S. Nature Conservation Drones for Automatic Localization and Counting of Animals. In Proceedings of the ECCV Workshops, Zurich, Switzerland, 6–7 September 2014; pp. 255–270.
14. Gonzalez, L.; Montes, G.; Puig, E.; Johnson, S.; Mengersen, K.; Gaston, K. Unmanned Aerial Vehicles (UAVs) and Artificial Intelligence Revolutionizing Wildlife Monitoring and Conservation. *Sensors* **2016**, *16*, 97. [CrossRef] [PubMed]
15. Bondi, E.; Jain, R.; Aggrawal, P.; Anand, S.; Hannaford, R.; Kapoor, A.; Piavis, J.; Shah, S.; Joppa, L.; Dilkina, B.; et al. BIRDSAI: A Dataset for Detection and Tracking in Aerial Thermal Infrared Videos. In Proceedings of the 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), Snowmass Village, CO, USA, 1–5 March 2020; pp. 1736–1745.
16. Chrétien, L.-P.; Théau, J.; Ménard, P. Visible and Thermal Infrared Remote Sensing for the Detection of White-Tailed Deer Using an Unmanned Aerial System: Detection of White-Tailed Deer Using an UAS. *Wildl. Soc. Bull.* **2016**, *40*, 181–191. [CrossRef]
17. Longmore, S.N.; Collins, R.P.; Pfeifer, S.; Fox, S.E.; Mulero-Pázmány, M.; Bezombes, F.; Goodwin, A.; De Juan Ovelar, M.; Knapen, J.H.; Wich, S.A. Adapting Astronomical Source Detection Software to Help Detect Animals in Thermal Images Obtained by Unmanned Aerial Systems. *Int. J. Remote Sens.* **2017**, *38*, 2623–2638. [CrossRef]
18. LeCun, Y.; Bengio, Y.; Hinton, G. Deep Learning. *Nature* **2015**, *521*, 436–444. [CrossRef]
19. He, K.; Zhang, X.; Ren, S.; Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1026–1034.
20. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
21. Christin, S.; Hervet, É.; Lecomte, N. Applications for Deep Learning in Ecology. *Methods Ecol. Evol.* **2019**, *10*, 1632–1644. [CrossRef]
22. Sun, C.; Shrivastava, A.; Singh, S.; Gupta, A. Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. *arXiv* **2017**, arXiv:1707.02968.
23. Maire, F.; Alvarez, L.M.; Hodgson, A. Automating Marine Mammal Detection in Aerial Images Captured During Wildlife Surveys: A Deep Learning Approach. In Proceedings of the Australasian Joint Conference on Artificial Intelligence, Canberra, ACT, Australia, 30 November–4 December 2015; pp. 379–385.
24. Gray, P.C.; Fleishman, A.B.; Klein, D.J.; McKown, M.W.; Bézy, V.S.; Lohmann, K.J.; Johnston, D.W. A Convolutional Neural Network for Detecting Sea Turtles in Drone Imagery. *Methods Ecol. Evol.* **2018**, *10*, 345–355. [CrossRef]
25. Gray, P.C.; Bierlich, K.C.; Mantell, S.A.; Friedlaender, A.S.; Goldbogen, J.A.; Johnston, D.W. Drones and Convolutional Neural Networks Facilitate Automated and Accurate Cetacean Species Identification and Photogrammetry. *Methods Ecol. Evol.* **2019**, *10*, 1490–1500. [CrossRef]
26. Bowler, E.; Fretwell, P.T.; French, G.; Mackiewicz, M. Using Deep Learning to Count Albatrosses from Space: Assessing Results in Light of Ground Truth Uncertainty. *Remote Sens.* **2020**, *12*, 2026. [CrossRef]
27. Smith, L.N. A Disciplined Approach to Neural Network Hyper-Parameters: Part 1—Learning Rate, Batch Size, Momentum, and Weight Decay. *arXiv* **2018**, arXiv:1803.09820.
28. Kellenberger, B.; Marcos, D.; Tuia, D. Detecting Mammals in UAV Images: Best Practices to Address a Substantially Imbalanced Dataset with Deep Learning. *Remote Sens. Environ.* **2018**, *216*, 139–153. [CrossRef]
29. Leevy, J.L.; Khoshgoftaar, T.M.; Bauder, R.A.; Seliya, N. A Survey on Addressing High-Class Imbalance in Big Data. *J. Big Data* **2018**, *5*, 42. [CrossRef]
30. Mazurowski, M.A.; Habas, P.A.; Zurada, J.M.; Lo, J.Y.; Baker, J.A.; Tourassi, G.D. Training Neural Network Classifiers for Medical Decision Making: The Effects of Imbalanced Datasets on Classification Performance. *Neutral Netw.* **2008**, *21*, 427–436. [CrossRef]
31. Chan, P.K.; Stolfo, S.J. Toward Scalable Learning with Non-Uniform Class and Cost Distributions: A Case Study in Credit Card Fraud Detection. In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98), New York, NY, USA, 27–31 August 1998; p. 12.
32. Mac Namee, B.; Cunningham, P.; Byrne, S.; Corrigan, O.I. The Problem of Bias in Training Data in Regression Problems in Medical Decision Support. *Artif. Intell. Med.* **2002**, *24*, 51–70. [CrossRef]
33. Buda, M.; Maki, A.; Mazurowski, M.A. A Systematic Study of the Class Imbalance Problem in Convolutional Neural Networks. *Neutral Netw.* **2018**, *106*, 249–259. [CrossRef]
34. Sung, K.-K.; Poggio, T. Example-Based Learning for View-Based Human Face Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1998**, *20*, 39–51. [CrossRef]

35. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*; Touretzky, D.S., Mozer, M.C., Hasselmo, M.E., Eds.; MIT Press: Cambridge, MA, USA, 2019; pp. 8026–8037.
36. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
37. Liu, L.; Jiang, H.; He, P.; Chen, W.; Liu, X.; Gao, J.; Han, J. On the Variance of the Adaptive Learning Rate and Beyond. *arXiv* **2020**, arXiv:1908.03265.
38. Zhang, M.R.; Lucas, J.; Hinton, G.; Ba, J. Lookahead Optimizer: K Steps Forward, 1 Step Back. *arXiv* **2019**, arXiv:1907.08610.
39. Prechelt, L. Early Stopping—But When? In *Neural Networks: Tricks of the Trade*; Lecture Notes in Computer Science; Orr, G.B., Müller, K.-R., Eds.; Springer: Berlin/Heidelberg, Germany, 1998; Volume 1524, pp. 55–69. ISBN 978-3-540-65311-0.
40. Smith, L.N. Cyclical Learning Rates for Training Neural Networks. *arXiv* **2015**, arXiv:1506.01186.
41. Chicco, D.; Jurman, G. The Advantages of the Matthews Correlation Coefficient (MCC) over F1 Score and Accuracy in Binary Classification Evaluation. *BMC Genom.* **2020**, *21*, 6. [CrossRef] [PubMed]
42. Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 2921–2929.
43. Norouzzadeh, M.S.; Nguyen, A.; Kosmala, M.; Swanson, A.; Palmer, M.S.; Packer, C.; Clune, J. Automatically Identifying, Counting, and Describing Wild Animals in Camera-Trap Images with Deep Learning. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, E5716–E5725. [CrossRef] [PubMed]
44. Willi, M.; Pitman, R.T.; Cardoso, A.W.; Locke, C.; Swanson, A.; Boyer, A.; Veldthuis, M.; Fortson, L. Identifying Animal Species in Camera Trap Images Using Deep Learning and Citizen Science. *Methods Ecol. Evol.* **2019**, *10*, 80–91. [CrossRef]