Two-phase training mitigates class imbalance for camera trap image classification with CNNs

Farjad Malik **Simon Wouters*** **Ruben Cartuyvels†**
ruben.cartuyvels@kuleuven.be

Erfan Ghadery
erfan.ghadery@kuleuven.be

Marie-Francine Moens
sien.moens@kuleuven.be

Department of Computer Science
KU Leuven

Abstract

By leveraging deep learning to automatically classify camera trap images, ecologists can monitor biodiversity conservation efforts and the effects of climate change on ecosystems more efficiently. Due to the imbalanced class-distribution of camera trap datasets, current models are biased towards the majority classes. As a result, they obtain good performance for a few majority classes but poor performance for many minority classes. We used two-phase training to increase the performance for these minority classes. We trained, next to a baseline model, four models that implemented a different versions of two-phase training on a subset of the highly imbalanced Snapshot Serengeti dataset. Our results suggest that two-phase training can improve performance for many minority classes, with limited loss in performance for the other classes. We find that two-phase training based on majority undersampling increases class-specific F1-scores up to 3.0%. We also find that two-phase training outperforms using only oversampling or undersampling by 6.1% in F1-score on average. Finally, we find that a combination of over- and undersampling leads to a better performance than using them individually.

1 Introduction

A recent report by the World Wide Fund for Nature (WWF) confirms that biodiversity and ecosystems are deteriorating worldwide [1]. Population sizes of mammals, birds, amphibians, reptiles and fish have decreased by an average of 68% between 1970 and 2016 across the world. This decrease in biodiversity has several causes, such as habitat loss due to pollution, species overexploitation or climate change. Biodiversity is important since it is a key indicator of overall healthy ecosystems which in their turn have important social and economic consequences for humans. In particular, biodiversity and ecosystems influence our water quality, air quality and climate, they secure our food production and impact the spread of infectious diseases originating from animals [1, 5].

Machine learning (ML) can help to more efficiently measure and monitor the well-being of ecosystems and the success of biodiversity conservation efforts [14, 16, 35, 25]. As an example, this paper proposes a method for automatic classification of camera trap images, a type of motion triggered cameras used in biological studies to estimate animal population density and activity patterns [26,

*Joint first author.
†Corresponding author.

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

7, 27, 29, 32, 34]. Since manually labeling large numbers of camera trap images is time consuming and costly [17], ML could be used to automatically detect animals and the species to which they belong in images. This work uses Convolutional Neural Networks [18, 19] to classify camera trap images. Training a CNN on a dataset of camera trap images is challenging, because camera trap images often only depict a part of an animal, because of high intra-class variation due to differences in backgrounds, and because the class-distribution of camera trap datasets is typically highly imbalanced. This imbalance is inherent to camera trap datasets since it reflects the imbalance of ecosystems [33], and it results in biased classifiers that perform very well for a few majority classes but poorly for many minority classes. Classifiers that perform well on all classes would be of more value to ecologists, and moreover, rare or less observed animal species might even be of special interest to research. Therefore, solutions are needed to mitigate this imbalance when classifying camera trap images.

To this end, we use a two-phase training method [20] to mitigate class imbalance, for the first time to the best of our knowledge on camera trap images. In experiments we compare it to different data-level class imbalance mitigation techniques, and show that it improves performance on minority classes, with limited loss in performance for other classes, resulting in an increase in macro F1-score.

2 Related work

Pioneering studies that automatically classified camera trap images relied on manual feature extraction and smaller datasets [6, 31, 39, 4]. Better and more scalable results were later achieved with deep CNNs and larger datasets [8, 24, 32, 38, 28]. Generally, models trained by these scholars achieve accuracies well above 90%, but the models are biased towards majority classes, which severely affects their class-specific performance. Especially the performance for rare species is poor. Scholars dealt with this challenge by removing the rare classes from the dataset [8, 38], with confidence thresholding and letting experts review the uncertain classifications [38], with weighted losses, oversampling and emphasis sampling [24] or by using a combination of additional image augmentations for rare classes and novel sampling techniques [28]. Although [24] managed to greatly increase the accuracy for a few rare classes using oversampling, none of the aforementioned techniques systematically improved accuracy for most of the rare species. It can thus be concluded that dealing with class-imbalance in the context of camera trap image classification is still an unsolved issue.

Two categories of methods for mitigation of class imbalance in deep learning exist: data-level and algorithm-level techniques [2, 15]. The former refers to techniques that alter the class-distribution of the data, such as random minority oversampling (ROS) and random majority undersampling (RUS), which respectively randomly duplicate or randomly remove samples to obtain a more balanced dataset. More advanced techniques can also be used to synthesize new samples [3, 9, 12, 36, 37, 21], but these are computationally expensive, and they require a large number of images per class and images within a class that are sufficiently similar. Algorithm-level techniques are techniques that work on the model itself, such as loss functions or thresholding [22, 23, 2, 15, 11, 2]. Two-phase training, a hybrid technique, was recently introduced and shown to obtain good results for training a CNN classifier on a highly imbalanced dataset of images of plankton [20], and it was later used by others for image segmentation and classification [10, 2]. Because of these promising results and the broad applicability of 2-phase training, we test 2-phase training for camera trap images.

3 Two-phase training

Two-phase training consists of the following steps [20]. $\mathcal{D}_{orig}$ is the original, imbalanced dataset. Figure 3 in the appendix shows an overview of two-phrase training.

1. **Phase 1**: a CNN f_θ is trained on a more balanced dataset $\mathcal{D}_{bal}$, obtained by any sampling method such as ROS, RUS or a combination thereof.
2. **Phase 2**: the convolutional weights[3] of f_θ are frozen, and the network is trained further on the full imbalanced dataset $\mathcal{D}_{org}$.

The 1st phase trains the convolutional layers with (more) equal importance allocated to minority classes, so they learn to extract relevant features for these classes as well. In the 2nd phase the classification layers learn to model the class imbalance present in the dataset.

[3] I.e. all weights except the weights of the fully connected layers that project the CNN features to the classes.

Model	Phase 1: Accuracy	Phase 2: Accuracy	Phase 1: F1	Phase 2: F1
$\mathcal{D}_{orig}$: Baseline	**0.8527**	/	0.3944	/
$\mathcal{D}_{bal}^1$: ROS	0.8326	**0.8528**	0.3843	0.4012
$\mathcal{D}_{bal}^2$: RUS	0.8012	0.8491	0.3681	**0.4147**
$\mathcal{D}_{bal}^3$: ROS&RUS(15K)	0.8346	0.8454	**0.4179**	0.4094
$\mathcal{D}_{bal}^4$: ROS&RUS(5K)	0.7335	0.8066	0.3620	0.4001

Table 1: Model Comparison - Top-1 accuracy and Macro F1-score.

4 Dataset & Experiments

We used the 9th season of the publicly available Snapshot Serengeti (SS) dataset, which is generated by a grid of 225 cameras spread over the Serengeti National Park in Tanzania [30]. The images were labeled by citizen scientists on the Zooniverse platform. After filtering some images, the full dataset $\mathcal{D}_{orig}$ contains 194k images belonging to 52 classes. The class-distribution of this dataset is depicted in fig. 4 in the appendix, and is highly imbalanced, with the three majority classes accounting for just under 75% of the data. We used this smaller subset of the full SS dataset for computational tractability, and to ensure insights remain valid for ecologists with access to smaller datasets.

Appendix A.2 lists the hyperparameters[4]. First we trained the baseline CNN on the full dataset $\mathcal{D}_{orig}$. Next, we trained 4 models with different instantiations of $\mathcal{D}_{bal}$ for phase 1 of two-phase training.

1. $\mathcal{D}_{bal}^1$: ROS (oversampling) classes with less than 5k images until 5k, see appendix fig. 5.
2. $\mathcal{D}_{bal}^2$: RUS (undersampling) classes with more than 15k images until 15k.
3. $\mathcal{D}_{bal}^3$: ROS classes with less than 5k images until 5k as in 1., and RUS classes with more than 15k images until 15k as in 2. Shown in fig. 6 in the appendix.
4. $\mathcal{D}_{bal}^4$: ROS classes with less than 5k images until 5k as in 1., and RUS classes with more than 5k images until 5k.

We used a lower sample ratio for classes with very few images to avoid overfitting (appendix A.3). As evaluation metrics we used not only top-1 accuracy but also precision, recall and F1-score, since these metrics are more informative to class-imbalance. We report their values macro-averaged over classes as well as the class specific values (in appendix tables 4-6). The results of the models after phase 1 correspond to the results that we would obtain by only using ROS, RUS or a combination of both (and no two-phase training). These results will thus serve as a baseline.

5 Results

Accuracy and Macro F1. Table 1 shows the accuracy and F1-score of the models after the 1st and the 2nd phase[5]. Training on more balanced datasets reduces accuracy in phase 1 for all models compared to the baseline which was trained on the imbalanced dataset $\mathcal{D}_{orig}$. However, further training the classification layers in phase 2 on the full dataset increases accuracy back to more or less the baseline level for all models (except ROS&RUS(5K)), meaning that two-phase training lost little to no accuracy. The phase 2 mean accuracy is substantially higher than the phase 1 mean accuracy.

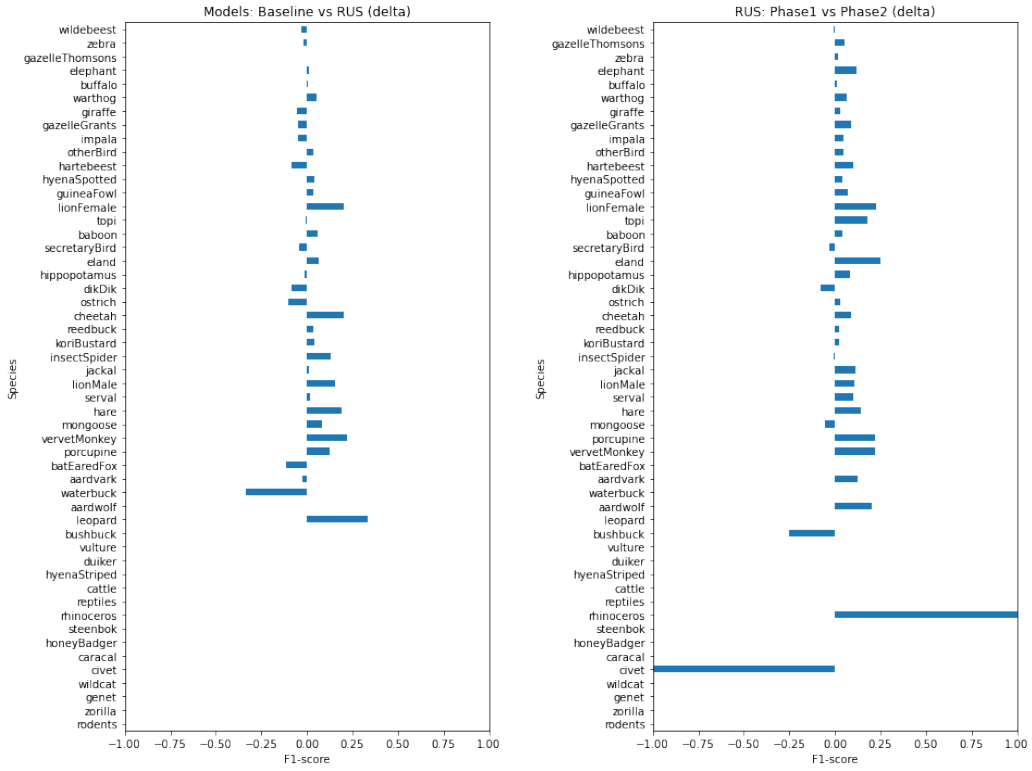
The F1-scores of most models also drop in phase 1. Interestingly, phase 2 raises the F1-score of most models again, and all models obtain an F1-score after phase 2 that is higher than the baseline: 3.0% on average. The RUS model obtains the highest F1-score after phase 2: an increase of 5.1% compared to the baseline, while the ROS&RUS(15K) model obtain the highest F1-score overall[6]. Most two-phase trained models outperform their counterparts which were only trained on more balanced datasets. As for the accuracy, the mean F1-score in phase 2 is substantially higher than the mean F1-score in phase 1: 6.1%.

These observations lead us to conclude that 1) two-phase training outperforms using only sampling techniques across most sampling regimes, and 2) two-phase training can increase the F1-score without

[4]Our code is publicly available: `https://github.com/FarjadMalik/aigoeswild`.

[5]Appendix A.4 contains more results and in-depth discussion.

[6]We consider the F1-score of ROS&RUS(15K) after phase 1 an anomaly which needs further analysis.



(a) (b)

Figure 1: Relative difference in F1-score per species of (a) the two-phase RUS model vs. the baseline, and (b) phase 2 vs. phase 1 of the RUS-model. The appendix contains larger versions: figs. 8, 10. Species are sorted in descending order according to their occurrence frequency.

substantial loss in accuracy, meaning it improves minority class predictions with very limited loss in majority class performance. These findings are in line with the results of [20], though they report greater increases in F1-scores than us, possibly due to an even more imbalanced dataset. They also find RUS to work best for creating $\mathcal{D}_{bal}$ for phase 1. The F1-scores are substantially lower than the accuracies (idem for precision and recall, appendix tables 2-3). This is because the class-specific values for these metrics are high for the majority classes, but extremely low for many minority classes, confirming that the imbalanced data creates a bias towards the majority classes.

Class-specific performance. Class-specific F1-scores increase with two-phase training for the majority of (minority) classes. Two-phase training with RUS leads to the greatest average increase of F1-score per class: 3% (ignoring the classes for which the F1-score remained 0.0%). This increase is most notable for minority classes. RUS performing best is remarkable, since we trained the RUS model in phase 1 with only 85k images, compared to 131k–231k for the other models. Fig. 1a shows the changes in F1-score due to two-phase training with RUS.

6 Conclusion

We explored the use of two-phase training to mitigate the class imbalance issue for camera trap image classification with CNNs. We conclude that 1) two-phase training outperforms using only sampling techniques across most sampling regimes, and 2) two-phase training improves minority class predictions with very limited loss in majority class performance, compared to training on the imbalanced dataset only. In the future we would like to rerun our experiments with different random seeds to obtain more statistically convincing results, compare two-phase training to other algorithm-level imbalance mitigation techniques, and test it on varying dataset sizes and levels of class imbalance.

References

[1] R. Almond, M. Grooten, and T. Peterson. Living planet report 2020-bending the curve of biodiversity loss. 2020.

[2] M. Buda, A. Maki, and M. A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

[3] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.*, 16:321–357, 2002.

[4] G. Chen, T. X. Han, Z. He, R. Kays, and T. Forrester. Deep convolutional neural network based species recognition for wild animal monitoring. In *2014 IEEE International Conference on Image Processing, ICIP 2014, Paris, France, October 27-30, 2014*, pages 858–862. IEEE, 2014.

[5] S. Díaz, J. Settele, E. S. Brondízio, H. T. Ngo, M. Guèze, J. Agard, A. Arneth, P. Balvanera, K. Brauman, S. H. Butchart, et al. Summary for policymakers of the global assessment report on biodiversity and ecosystem services of the intergovernmental science-policy platform on biodiversity and ecosystem services. *Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services*, 2019.

[6] K. Figueroa, A. Camarena-Ibarrola, J. Garcia, and H. T. Villela. Fast automatic detection of wildlife in images from trap cameras. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications - 19th Iberoamerican Congress, CIARP 2014, Puerto Vallarta, Mexico, November 2-5, 2014. Proceedings*, volume 8827 of *Lecture Notes in Computer Science*, pages 940–947. Springer, 2014.

[7] R. J. Foster and B. J. Harmsen. A critique of density estimation from camera-trap data. *The Journal of Wildlife Management*, 76(2):224–236, 2012.

[8] A. Gómez-Villa, A. Salazar, and J. F. Vargas-Bonilla. Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. *Ecol. Informatics*, 41:24–32, 2017.

[9] H. Han, W. Wang, and B. Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *Advances in Intelligent Computing, International Conference on Intelligent Computing, ICIC 2005, Hefei, China, August 23-26, 2005, Proceedings, Part I*, volume 3644 of *Lecture Notes in Computer Science*, pages 878–887. Springer, 2005.

[10] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. C. Courville, Y. Bengio, C. Pal, P. Jodoin, and H. Larochelle. Brain tumor segmentation with deep neural networks. *Medical Image Anal.*, 35:18–31, 2017.

[11] H. He and Y. Ma. *Imbalanced learning: foundations, algorithms, and applications*. Wiley-IEEE Press, 2013.

[12] H. He, Y. Bai, E. A. Garcia, and S. Li. ADASYN: adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN 2008, part of the IEEE World Congress on Computational Intelligence, WCCI 2008, Hong Kong, China, June 1-6, 2008*, pages 1322–1328. IEEE, 2008.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[14] D. N. T. Huynh and N. Neptune. Automatic image annotation : the case of deforestation. In *Actes de la Conférence CORIA-TALN-RJC - Volume 2 - Démonstrations, articles des Rencontres Jeunes Chercheurs, ateliers DeFT, Rennes, France, May 14-18, 2018*, pages 101–116. ATALA, 2018.

[15] J. M. Johnson and T. M. Khoshgoftaar. Survey on deep learning with class imbalance. *J. Big Data*, 6:27, 2019.

[16] A. Joly, H. Goëau, S. Kahl, C. Botella, R. L. R. D. Castaneda, H. Glotin, E. Cole, J. Champ, B. Deneu, M. Servajean, T. Lorieul, W. Vellinga, F. Stöter, A. Durso, P. Bonnet, and H. Müller. Lifeclef 2020 teaser: Biodiversity identification and prediction challenges. In *Advances in Information Retrieval - 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14-17, 2020, Proceedings, Part II*, volume 12036 of *Lecture Notes in Computer Science*, pages 542–549. Springer, 2020.

[17] M. J. Kelly, A. J. Noss, M. S. Di Bitetti, L. Maffei, R. L. Arispe, A. Paviolo, C. D. De Angelo, and Y. E. Di Blanco. Estimating puma densities from camera trapping across three study sites: Bolivia, argentina, and belize. *Journal of mammalogy*, 89(2):408–418, 2008.

[18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[19] Y. LeCun, Y. Bengio, and G. E. Hinton. Deep learning. *Nat.*, 521(7553):436–444, 2015.

[20] H. Lee, M. Park, and J. Kim. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In *2016 IEEE International Conference on Image Processing, ICIP 2016, Phoenix, AZ, USA, September 25-28, 2016*, pages 3713–3717. IEEE, 2016.

[21] Y. Li, J. Guo, X. Guo, Z. Hu, and Y. Tian. Plankton detection with adversarial learning and a densely connected deep learning model for class imbalanced distribution. *Journal of Marine Science and Engineering*, 9(6):636, 2021.

[22] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. volume 42, pages 318–327, 2020.

[23] K. Nemoto, R. Hamaguchi, T. Imaizumi, and S. Hikosaka. Classification of rare building change using CNN with multi-class focal loss. In *2018 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2018, Valencia, Spain, July 22-27, 2018*, pages 4663–4666. IEEE, 2018.

[24] M. S. Norouzzadeh, A. Nguyen, M. Kosmala, A. Swanson, M. S. Palmer, C. Packer, and J. Clune. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proc. Natl. Acad. Sci. USA*, 115(25):E5716–E5725, 2018.

[25] J. Park, J. Lee, K. Seto, T. Hochberg, B. A. Wong, N. A. Miller, K. Takasaki, H. Kubota, Y. Oozeki, S. Doshi, et al. Illuminating dark fishing fleets in north korea. *Science advances*, 6 (30):eabb1197, 2020.

[26] M. S. Ridout and M. Linkie. Estimating overlap of daily activity patterns from camera trap data. *Journal of Agricultural, Biological, and Environmental Statistics*, 14(3):322–337, 2009.

[27] J. M. Rowcliffe, R. Kays, B. Kranstauber, C. Carbone, and P. A. Jansen. Quantifying levels of animal activity using camera trap data. *Methods in Ecology and Evolution*, 5(11):1170–1179, 2014.

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

[29] R. Sollmann. A gentle introduction to camera-trap data analysis. *African Journal of Ecology*, 56(4):740–749, 2018.

[30] A. Swanson, M. Kosmala, C. Lintott, R. Simpson, A. Smith, and C. Packer. Snapshot serengeti, high-frequency annotated camera trap images of 40 mammalian species in an african savanna. *Scientific data*, 2(1):1–14, 2015.

[31] K. R. Swinnen, J. Reijniers, M. Breno, and H. Leirs. A novel method to reduce time investment when processing videos from camera trap studies. *PloS one*, 9(6):e98881, 2014.

[32] M. A. Tabak, M. S. Norouzzadeh, D. W. Wolfson, S. J. Sweeney, K. C. VerCauteren, N. P. Snow, J. M. Halseth, P. A. Di Salvo, J. S. Lewis, M. D. White, et al. Machine learning to classify animal species in camera trap images: Applications in ecology. *Methods in Ecology and Evolution*, 10(4):585–590, 2019.

[33] R. Trebilco, J. K. Baum, A. K. Salomon, and N. K. Dulvy. Ecosystem ecology: size-based constraints on the pyramids of life. *Trends in ecology & evolution*, 28(7):423–431, 2013.

[34] F. Trolliet, C. Vermeulen, M.-C. Huynen, and A. Hambuckers. Use of camera traps for wildlife studies: a review. *Biotechnologie, Agronomie, Société et Environnement*, 18(3):446–454, 2014.

[35] J. C. van Gemert, C. R. Verschoor, P. Mettes, K. Epema, L. P. Koh, and S. A. Wich. Nature conservation drones for automatic localization and counting of animals. In *Computer Vision - ECCV 2014 Workshops - Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part I*, volume 8925 of *Lecture Notes in Computer Science*, pages 255–270. Springer, 2014.

[36] Z. Wan, Y. Zhang, and H. He. Variational autoencoder based synthetic data generation for imbalanced learning. In *2017 IEEE Symposium Series on Computational Intelligence, SSCI 2017, Honolulu, HI, USA, November 27 - Dec. 1, 2017*, pages 1–7. IEEE, 2017.

[37] C. Wang, Z. Yu, H. Zheng, N. Wang, and B. Zheng. Cgan-plankton: Towards large-scale imbalanced class generation and fine-grained classification. In *2017 IEEE International Conference on Image Processing, ICIP 2017, Beijing, China, September 17-20, 2017*, pages 855–859. IEEE, 2017.

[38] M. Willi, R. T. Pitman, A. W. Cardoso, C. Locke, A. Swanson, A. Boyer, M. Veldthuis, and L. Fortson. Identifying animal species in camera trap images using deep learning and citizen science. *Methods in Ecology and Evolution*, 10(1):80–91, 2019.

[39] X. Yu, J. Wang, R. Kays, P. A. Jansen, T. Wang, and T. S. Huang. Automated identification of animal species in camera trap images. *EURASIP J. Image Video Process.*, 2013:52, 2013.

A Appendix

A.1 Figures

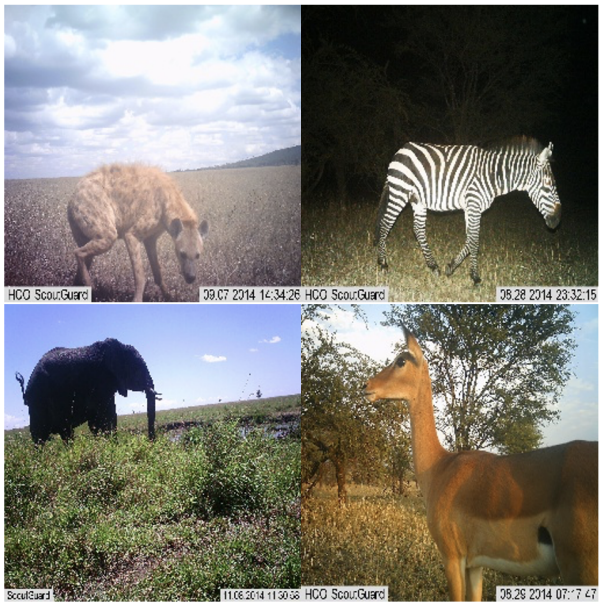


Figure 2: Four examples of camera trap images.

A.2 Experiments & hyperparameters

We omitted images that contain more than one species, images that do not contain any animal (class 'blank'), as well as images belonging to the class 'human'. We split our dataset following a 80%-10%-10% train, validation and test split. We chose for the ResNet-18 architecture [13], which performed well in previous camera trap literature [24, 32, 38]. We used the ADAM optimizer, a batch-size of 64 and a learning rate of 0.001. Data augmentation and early stopping (for the baseline as well as for both phases of all 2-phase models) were used to avoid overfitting. We used categorical cross-entropy as loss function. We trained the baseline model for 10 epochs on the training set, and the ROS model, the RUS model, the ROS&RUS(15K) model and the ROS&RUS(5K) model in the first phase for respectively 10, 14, 14 and 17 epochs. In the second phase, they were trained for an additional 7, 14, 8 and 14 epochs. Hyperparameters, incl. the ROS and RUS thresholds of 5k and 15k, were taken from existing literature, or experimentally tuned using a grid search.

After training all the four models in the first phase, we extracted the weights of all the 18 layers of the models. We loaded these weights into four new models with the same ResNet-18 architecture. Before fine-tuning these models on the original, imbalanced, training data set, we froze the weights in the 17 convolutional layers. Only the single fully connected layer of the ResNet-18 model was fine-tuned on the training set. By freezing the majority of the layers, the number of trainable parameters was reduced from slightly over 11 million to only 26,676. This implies that training the second phase is far less computationally demanding than training the models in the first phase.

A.3 Sampling

A.3.1 ROS

For the ROS data set, we oversampled all the classes that contained less than 5000 images to a maximum of 5000 images. The three majority classes (see fig. 4) were not oversampled. Since our minority classes contained very little samples, we did not oversample these classes up to 5000 images, as this would likely result in overfitting. Instead, for all classes that contained less than 100 images, we applied a sample ratio of 10. This means that the image(s) for a certain class were

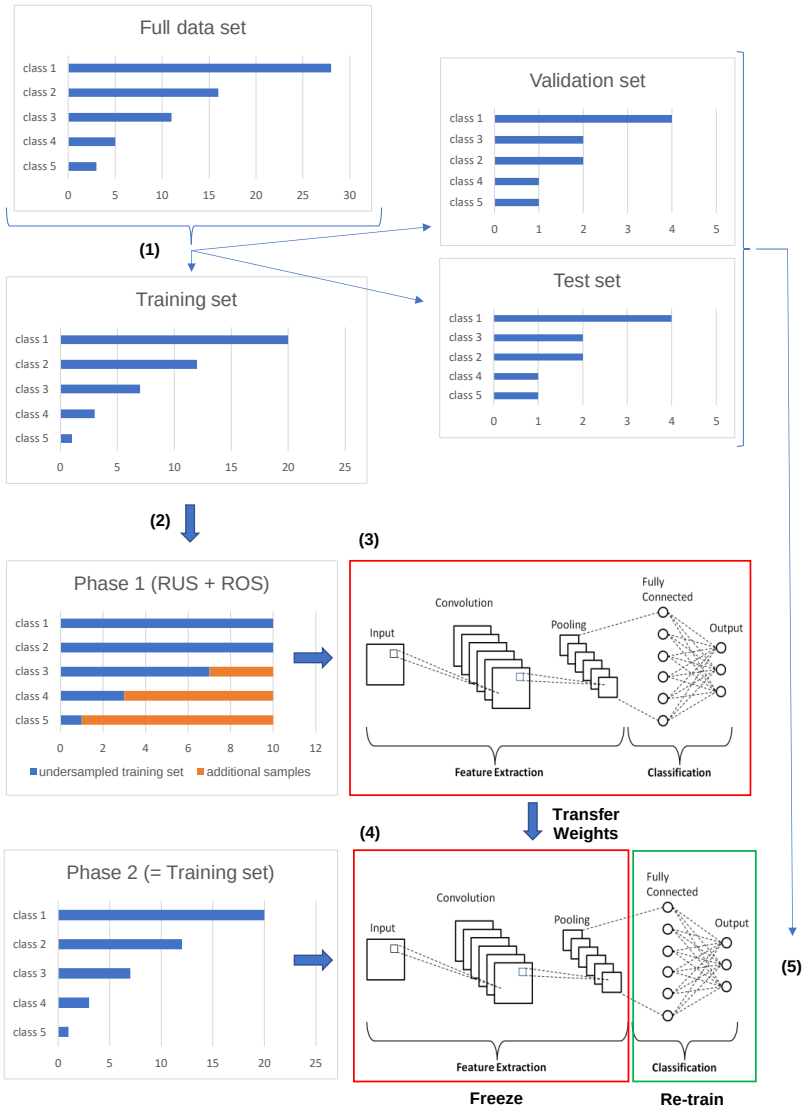


Figure 3: Schematic overview of a general two-phase training implementation.

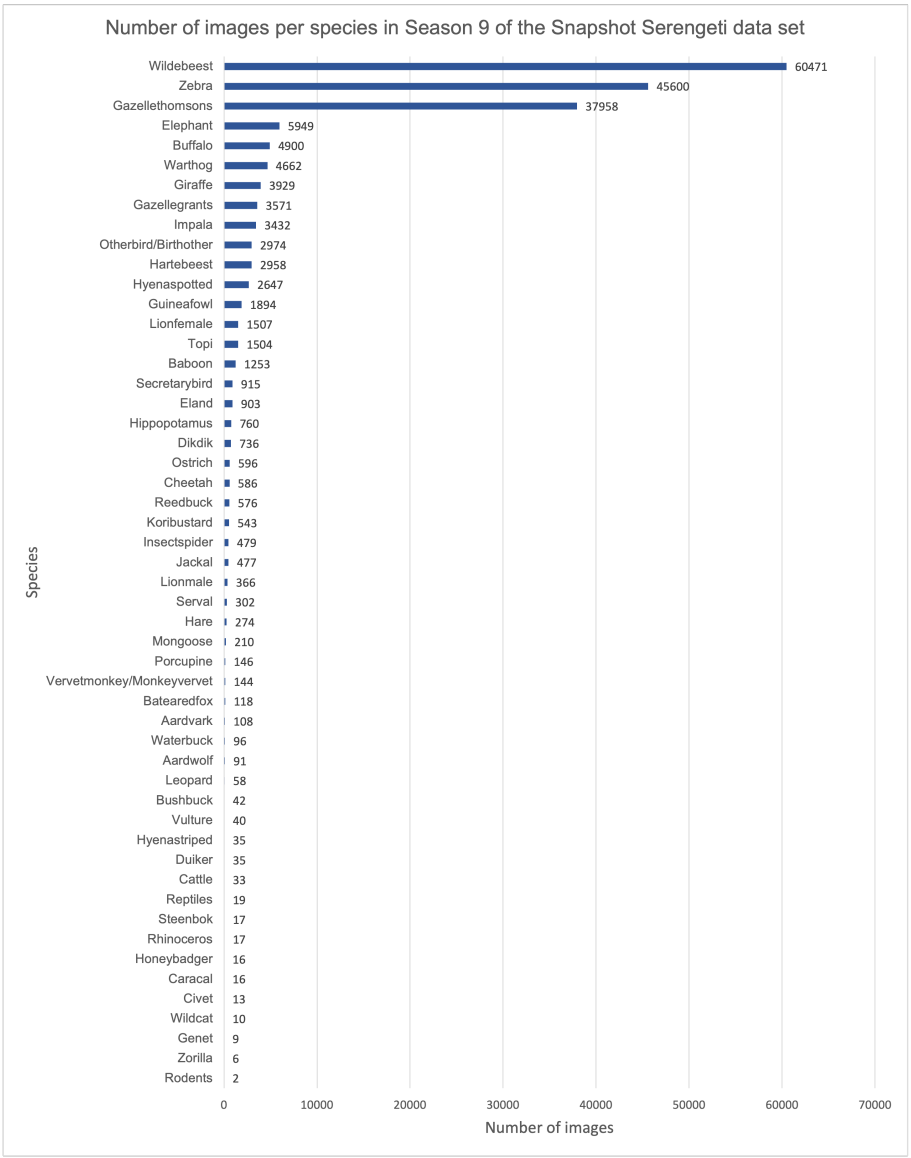


Figure 4: Class-distribution of the 9th season of the SS dataset.

oversampled until 10 times the original number of images was present in the data set. This is similar to Schneider et al. [28], who supplemented the minority classes with less than 100 images with fixed augmentations until at least 100 images were available. For their data set, this also implied a sampling ratio of roughly 10 for the smallest classes.

In order to obtain a more balanced dataset, this sampling ratio was gradually decreased. For classes that originally contained between 100 & 500, 500 & 1000, and 1000 & 5000, we used sample ratios of respectively 8, 6 and 4. We thresholded oversampling at 5000 for the classes that contained less than 5000 images. This threshold was set experimentally and is higher than the threshold of 1000 that Lee et al. [20] used when trying two-phase training in combination with ROS. We found that oversampling to 1000 images was not sufficient for our data set. It is, however, to be noted that a threshold of 5000 implies increased computational requirements. While the original training set contained slightly over 155,000 images, the ROS data set contained slightly over 231,000 images.

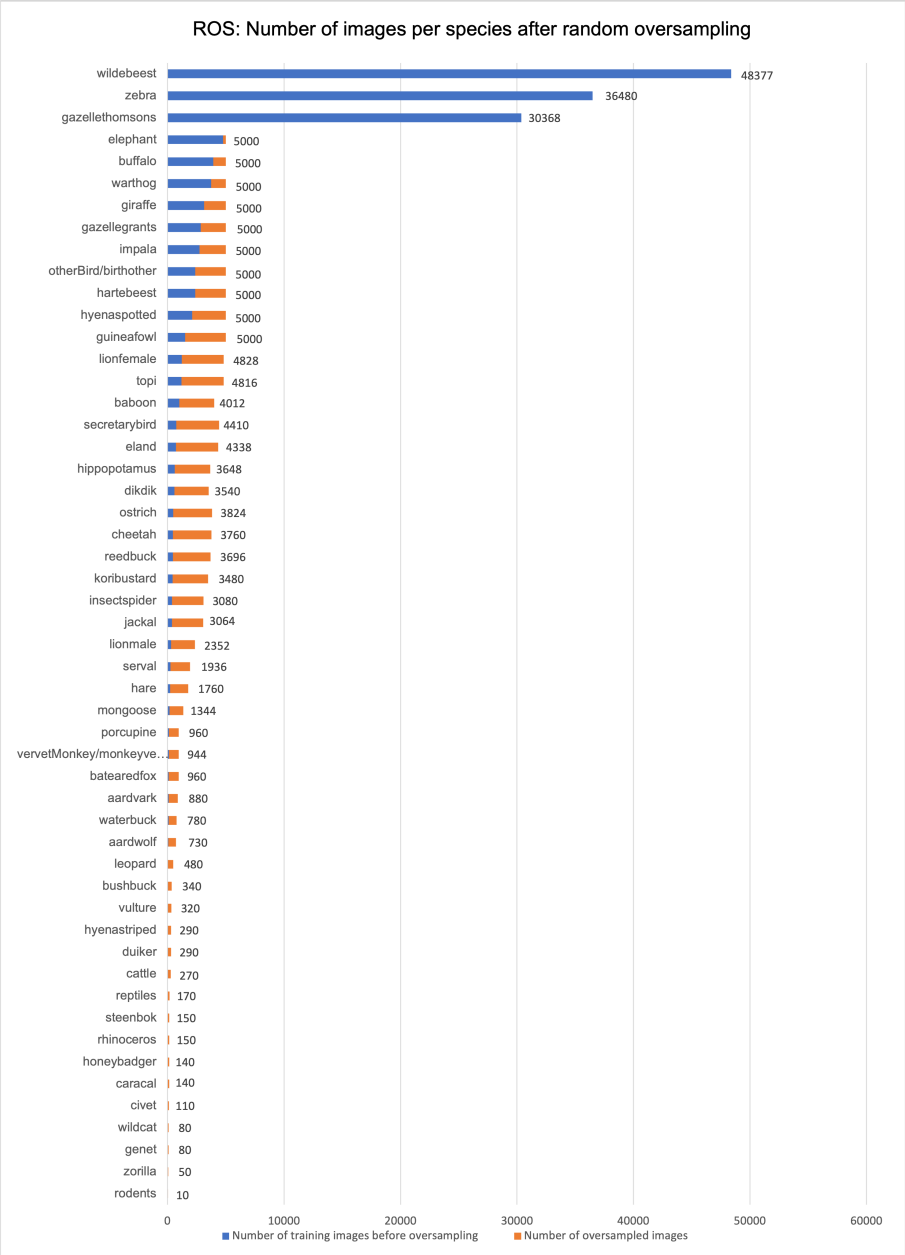


Figure 5: Class-distribution of the dataset used in the first phase for the ROS model.

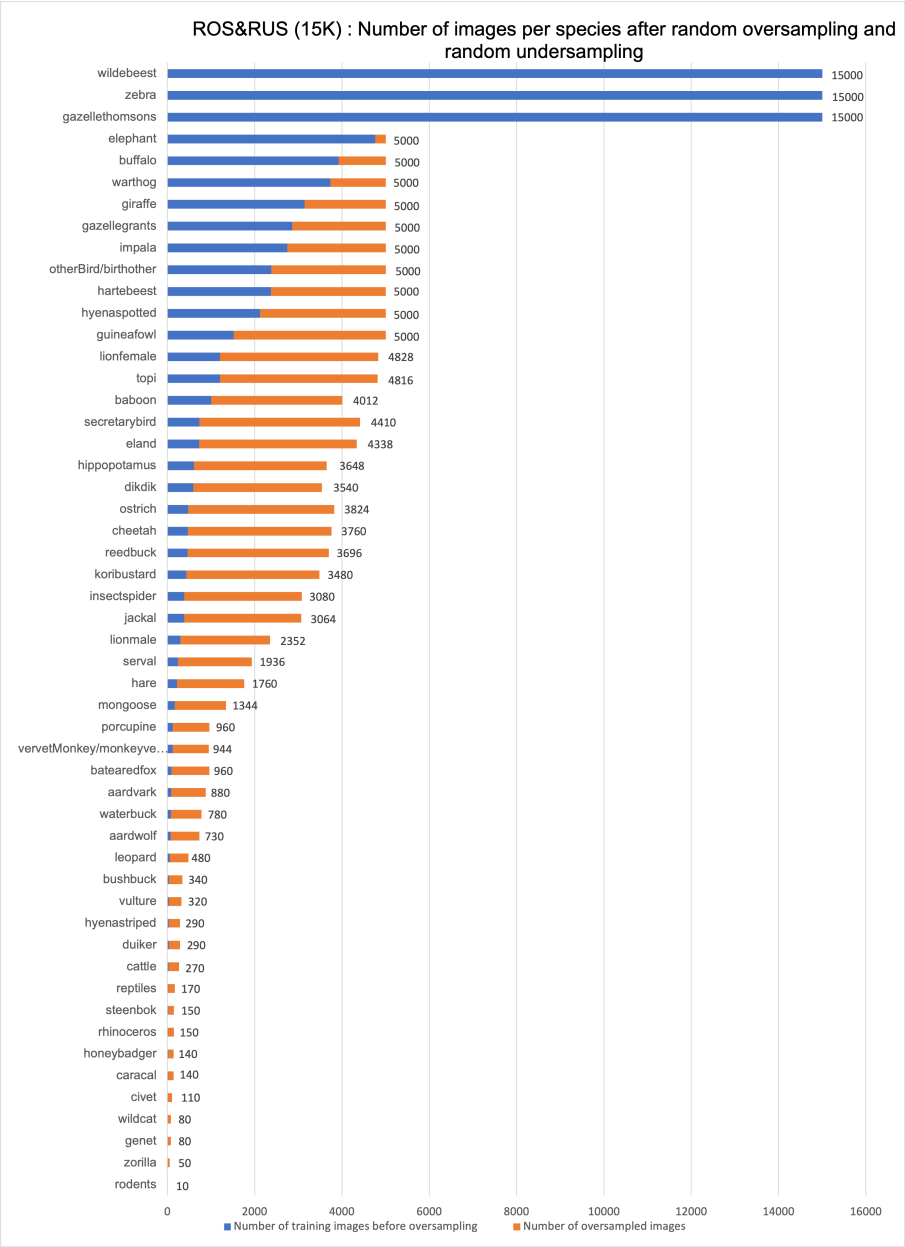


Figure 6: Class-distribution of the dataset used in the first phase for the ROS&RUS(15K) model.

Model	First Phase	Second Phase
Baseline	0.5055	N.A.
ROS	**0.4691**	**0.5053**
RUS	**0.4609**	**0.5319**
ROS&RUS (15K)	0.4519	0.5284
ROS&RUS (5K)	0.3702	0.5640

Table 2: Model Comparison - Precision.

Model	First Phase	Second Phase
Baseline	0.3558	N.A.
ROS	**0.3689**	**0.3635**
RUS	**0.3592**	**0.3648**
ROS&RUS (15K)	0.4094	0.3563
ROS&RUS (5K)	0.3974	0.3438

Table 3: Model Comparison - Recall.

A.3.2 RUS

To avoid an increase in the number of samples in our dataset, such as in the case of ROS, we decided to also train a model on a randomly undersampled dataset. Lee et al. [20] obtained the best results for two-phase training when using RUS only. The authors put their threshold at 5000, meaning that all the classes with more than 5000 images were undersampled until they held at most 5000 samples. Experimental results indicated that this threshold was too low for us when it was not used in combination with ROS. The vast reduction in the size of the data set might be a possible explanation for this. Therefore, we set the undersampling threshold at 15,000 for the RUS model. The RUS data set thus looks exactly the same as the original training data set, except for the fact that the three majority classes all contain 15,000 images instead of respectively 48,377; 36,480; and 30,368 images. This brought the total number of images for the RUS data set to slightly over 85,000.

A.3.3 ROS&RUS(15K) and ROS&RUS(5K)

The limited previous literature on two-phase training has not used this method in combination with both ROS and RUS at the same time in the first phase. Therefore, we decided to explore the effectiveness of two-phase training when both ROS and RUS are used in the first phase. For this purpose, we created two more balanced data sets, which we refer to as ROS&RUS(15K) and ROS&RUS(5K). We obtained the ROS&RUS(15K) data set by combining the over- and undersampling regimes of the ROS model and the RUS model respectively. The class-distribution of this data set is depicted in figure 6. Interestingly, by using these two sampling regimes, we got a data set that only contains 6000 images more than the original training set.

A.4 Extra results & discussion

A.4.1 Overall performance of the models: Overall top-1 accuracy and & Macro F1-score

Table 1 showed the top-1 accuracy and the Macro F1-score of the models for the 1st and the 2nd phase. Tables 2 and 3 additionally show the precision and recall.

The values of the class-weighted F1-score (same for precision and recall as shown in appendix table 2 and 3) are substantially lower than our overall accuracy. This is caused by the fact that the class-specific values for these metrics are very high for the majority classes, but extremely low for many minority classes. For example, the recall varies between 0% for the minority classes and 96.0% for the majority class. Many of these minority classes contain only few images in the training and the test set, which explains why the model performs poorly on them. This indicates that the imbalanced data indeed created a bias towards the majority class(es). This bias is also reflected in the difference in discrepancy between precision and recall for classes of different sizes. For the majority classes, we find a higher recall than precision. This indicates that the model predicts the majority class more

often than necessary. For the minority class, the precision tends to be higher than the recall. This indicates that the baseline model could likely improve performance when more attention is diverted from the majority classes to the minority classes.

Taking both overall accuracy and macro F1-score into account, we can conclude that two-phase training can result in a small increase in F1-score without losing overall accuracy. This slight increase in F1-score can be attributed to a slight increase of both the precision and recall. The ROS and the RUS models seem to perform best, given that they improve upon the F1-score of the baseline model, without or with little decrease in overall accuracy. Generally, two-phase training also outperforms only using ROS, RUS or a combination of both. Training in two phases leads to an improvement in both overall accuracy and F1-score compared to the models that were trained in one phase on a more balanced dataset. An exception to this observation is the ROS&RUS(15K) model. This model achieves the highest F1-score but performs slightly worse than the baseline model in terms of overall accuracy. From our results, we can see that the RUS model benefits more in terms of increase in precision when fine-tuning the classification layer. Possibly, training the ROS model for both phases with full majority classes limited the increase in precision for this model, since a focus on majority classes tends to lead to many false positives and thus a lower precision.

Our results are in accordance with findings of Lee et al. [20], who also concluded that two-phase training in combination with RUS leads to the highest increase in F1-score with respect to the baseline model. However, they managed to almost double their baseline F1-score from 17.73% to 33.39%. One possible explanation for this larger increase in F1-score might be that their dataset was more imbalanced than ours. This would mean that two-phase training becomes more effective when datasets become more imbalanced. Nevertheless, this thesis thus presents some evidence for the argument by Lee et al. [20], who stated that by making the dataset more balanced in the first phase, animal population information is lost and that this information consequently needs to be restored by fine-tuning the classification layer on the training set representing the population distribution of the ecosystem.

A.4.2 Class-specific performance

Table 5 and table 6 show respectively the class-specific results for the ROS and the RUS model, when trained using two-phase training. These tables follow the same structure as table 4, which shows the class-specific results for the baseline model, except for the fact that the Count(Train) column now depicts the number of samples that were used for training in the first phase only. In the second phase, all the four models were fine-tuned on the original training set. Figs. 7 and 8 show the relative change in F1-score when using ROS and RUS, respectively, versus the baseline.

A.4.3 Discussion of two-phase training: phase 1 compared to phase 2

Figs. 9 and 10 show the relative change in F1-score when using ROS and RUS, respectively, in phase 2 versus phase 1.

The large performance differences between the first and the second phase of the ROS, the RUS and the ROS&RUS(5K) model can mainly be attributed to the vast increase in precision that is obtained when fine-tuning the classification layer. A possible explanation for this observation is that in the first phase, precision and recall are low given that the distribution of the training set does not match that one of the test set. At this stage, the model might be good at feature extraction for the minority class images. However, the classification layer is still not used to the real data distribution. Therefore, the models might predict the oversampled classes too often, leading to a lower precision. After fine-tuning the classification layer, the model might be better able to combine its enhanced feature extraction skills with respect to the minority class images with the knowledge it has on the real data distribution.

A.5 Limitations & future work

We did not achieve a similar overall accuracy for our baseline model compared to most other works. Although achieving a high overall accuracy was not the main goal of this thesis, we consider our baseline performance as a limitation since it remains unknown whether two-phase training could also lead to increase in F1-score without losing overall accuracy, when the accuracy is very high.

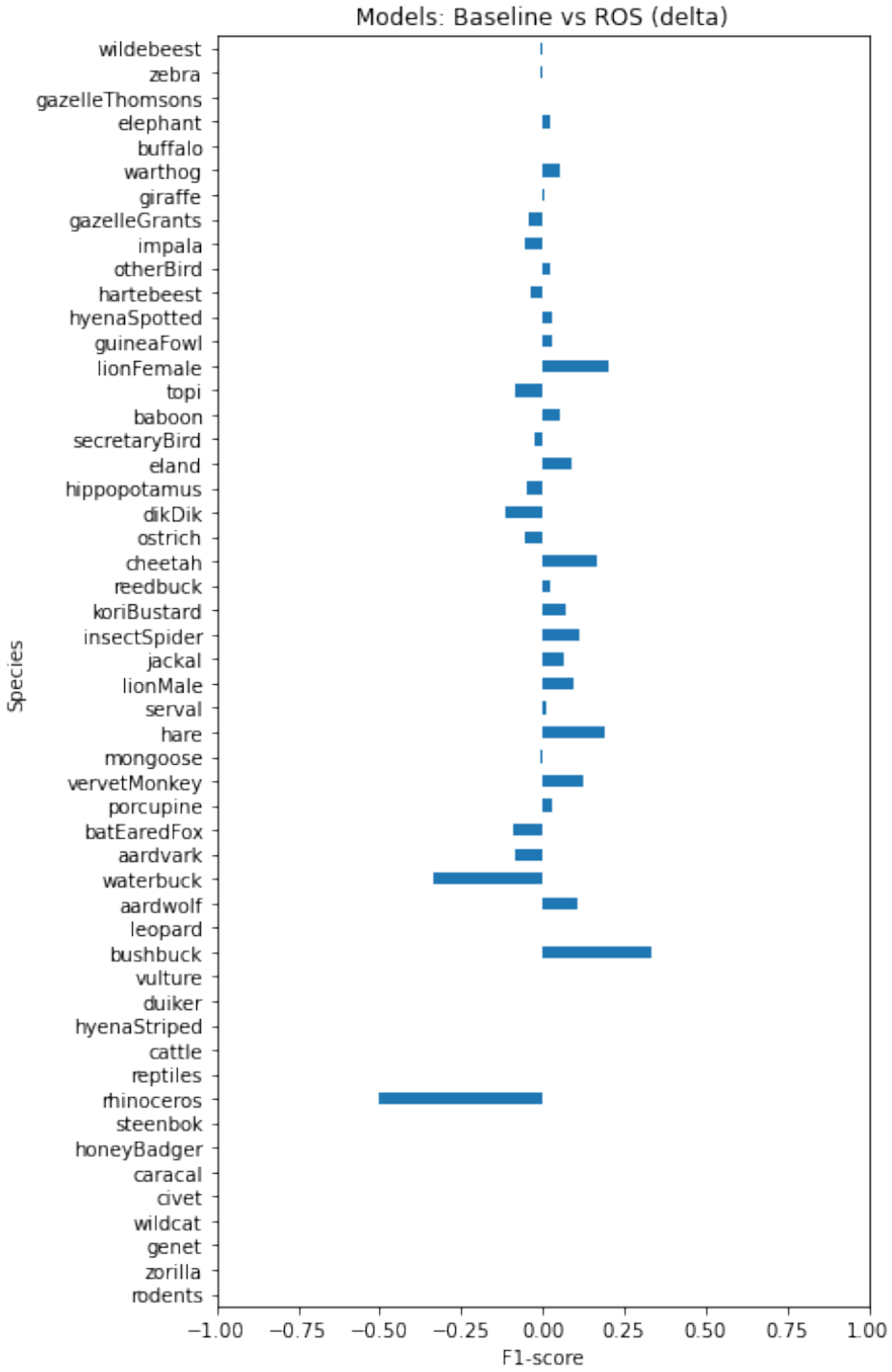


Figure 7: Percentage increase or decrease in F1-score per species of the ROS model compared to the baseline model.

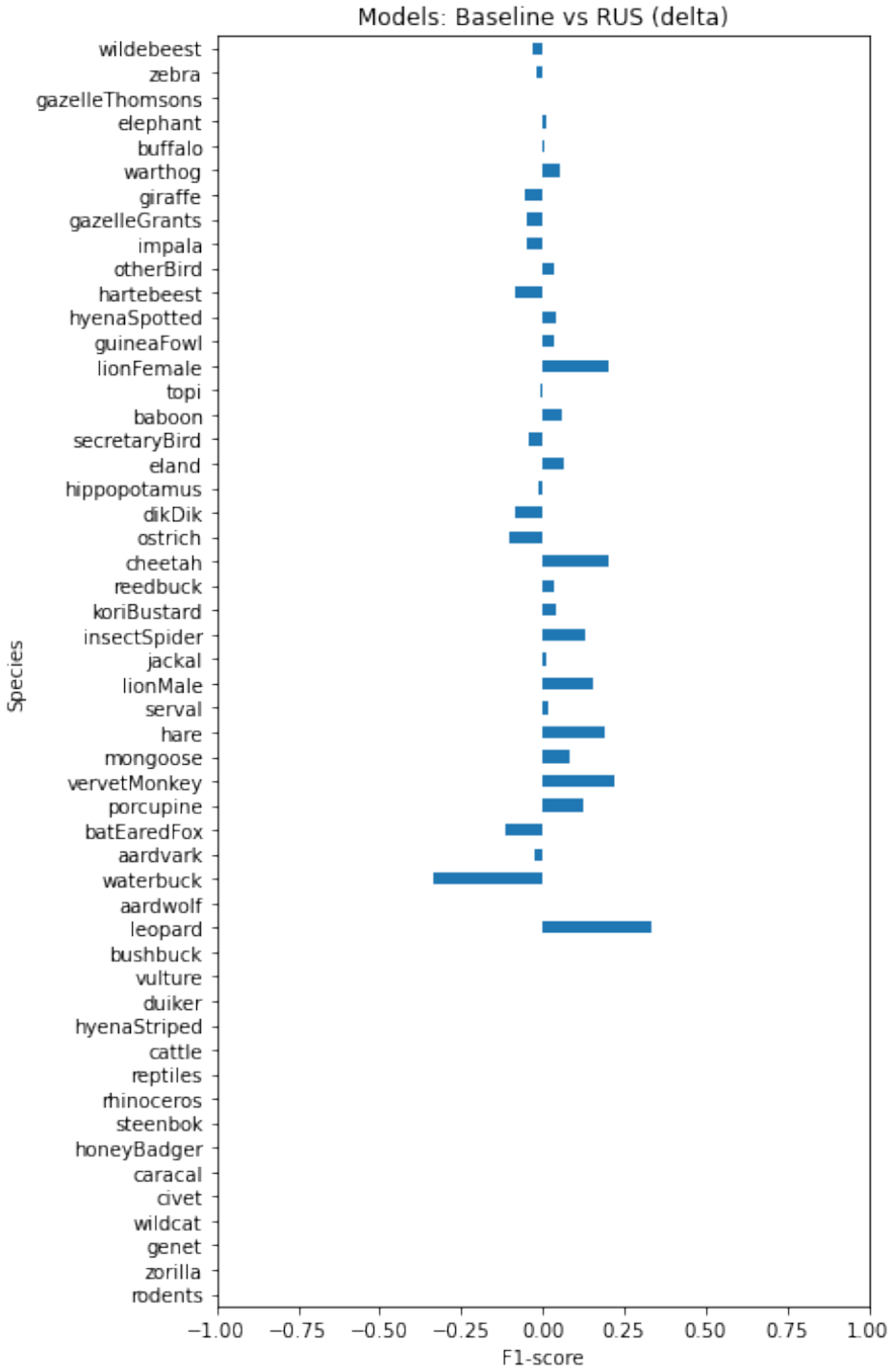


Figure 8: Percentage increase or decrease in F1-score per species of the RUS model compared to the baseline model.

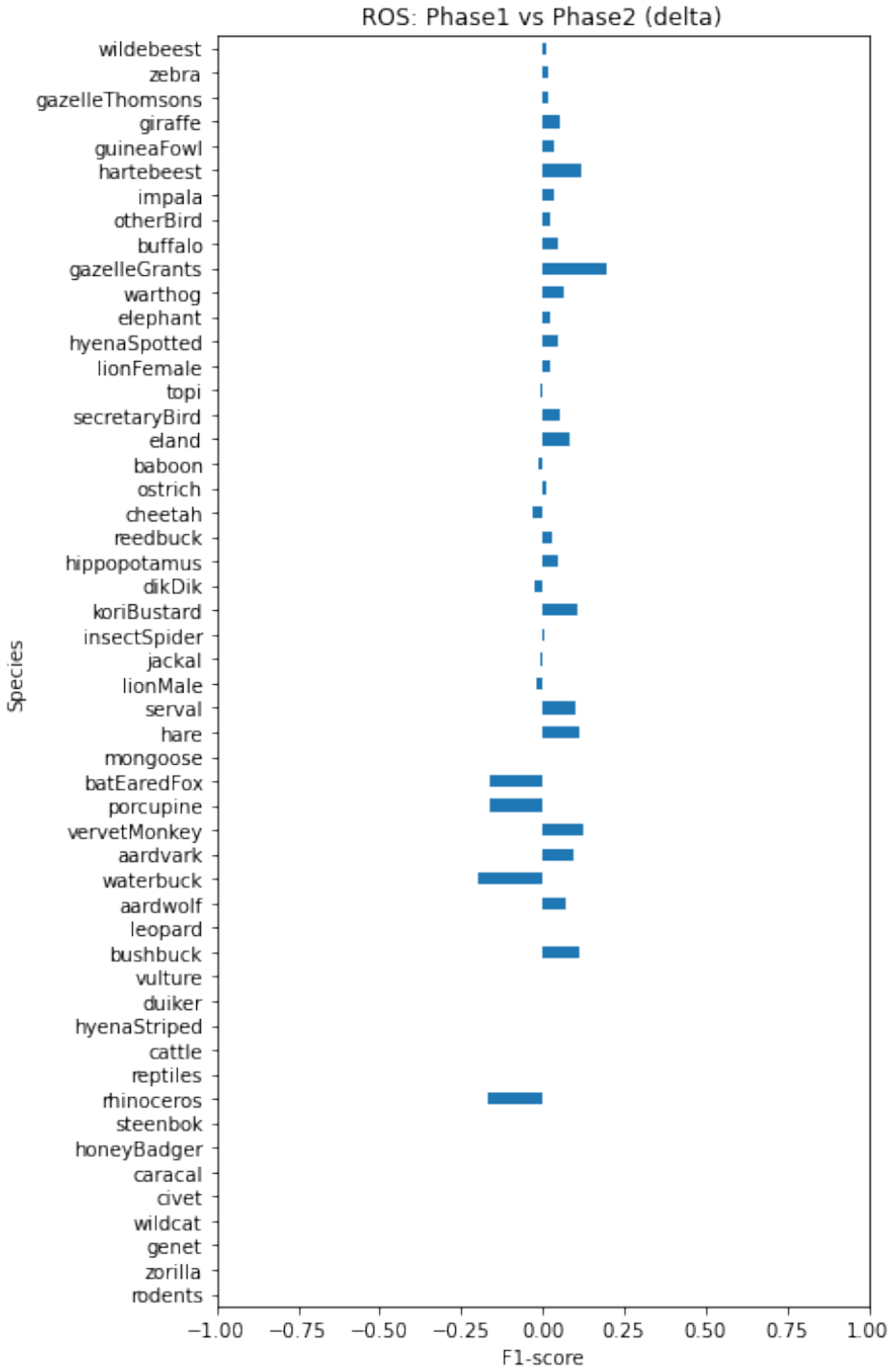


Figure 9: Percentage increase or decrease in F1-score per species of the second phase of the ROS model compared to the first phase.

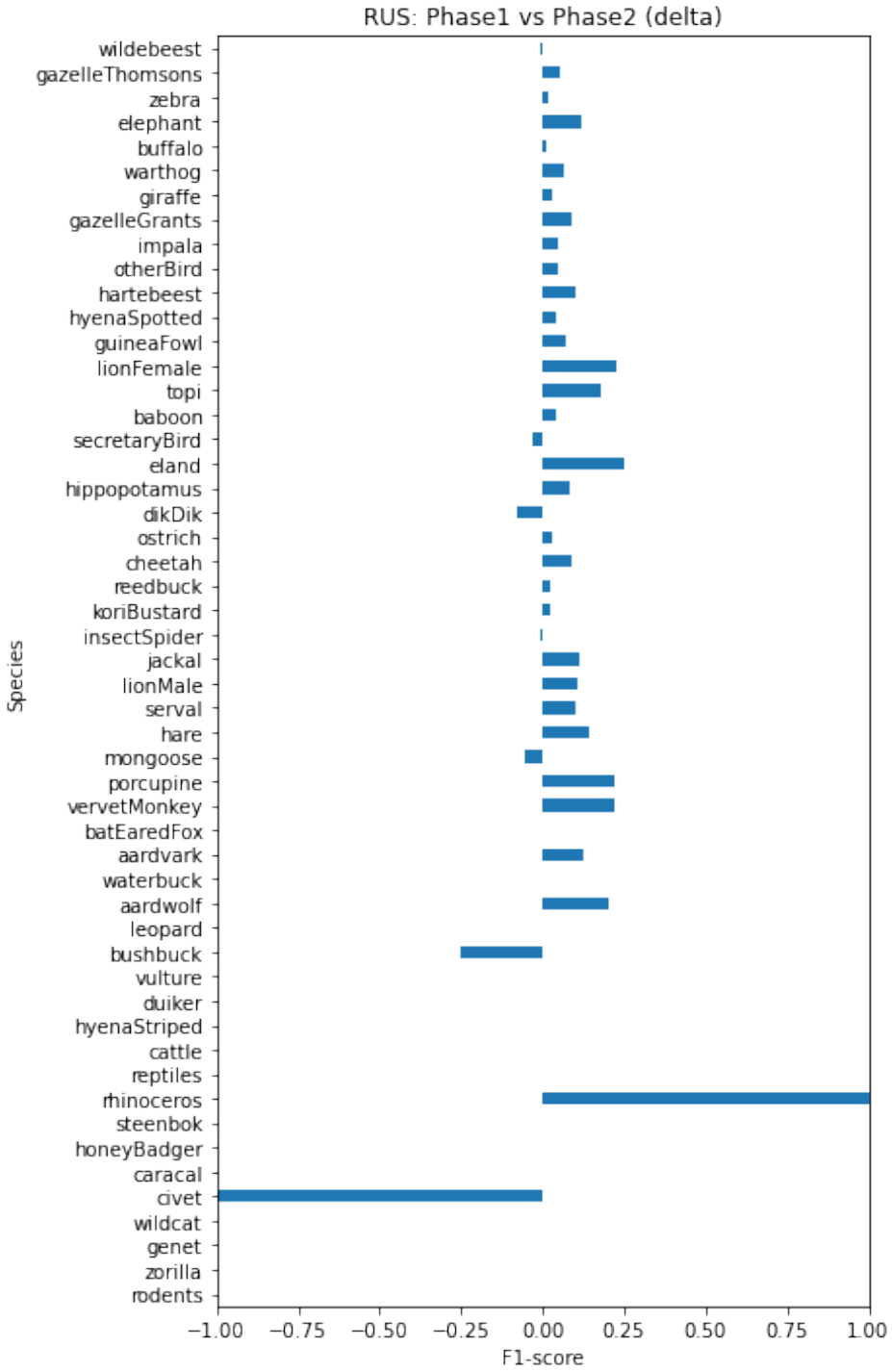


Figure 10: Percentage increase or decrease in F1-score per species of the second phase of the RUS model compared to the first phase.

Species	Precision	Recall	F1-score	Count (Train)	Count (Test)
wildebeest	0.8612	0.9601	0.908	48377	6047
zebra	0.9096	0.9377	0.9234	36480	4560
gazelleThomsons	0.9116	0.9078	0.9097	30368	3795
elephant	0.9382	0.6902	0.7953	4761	594
buffalo	0.804	0.6531	0.7207	3920	490
warthog	0.5548	0.6631	0.6041	3730	466
giraffe	0.9286	0.7628	0.8375	3145	392
gazelleGrants	0.5641	0.4314	0.4889	2857	357
impala	0.7595	0.7551	0.7573	2746	343
otherBird	0.5246	0.431	0.4732	2380	297
hartebeest	0.685	0.6339	0.6585	2368	295
hyenaSpotted	0.5202	0.6326	0.5709	2119	264
guineaFowl	0.7963	0.6825	0.735	1516	189
lionFemale	0.9545	0.28	0.433	1207	150
topi	0.5764	0.5533	0.5646	1204	150
baboon	0.7021	0.528	0.6027	1003	125
secretaryBird	0.9012	0.8111	0.8538	735	90
eland	0.9429	0.3667	0.528	723	90
hippopotamus	0.9143	0.8421	0.8767	608	76
dikDik	0.6747	0.7671	0.7179	590	73
ostrich	0.6667	0.339	0.4494	478	59
cheetah	0.4615	0.2069	0.2857	470	58
reedbuck	0.7292	0.614	0.6667	462	57
koriBustard	0.4677	0.537	0.5	435	54
insectSpider	0.3636	0.0851	0.1379	385	47
jackal	0.4583	0.234	0.3099	383	47
lionMale	0.375	0.1667	0.2308	294	36
serval	0.6	0.3	0.4	242	30
hare	0.9231	0.4444	0.6	220	27
mongoose	0.4	0.1905	0.2581	168	21
vervetMonkey	0.0	0.0	0.0	120	12
porcupine	1.0	0.2857	0.4444	118	14
batEaredFox	0.75	0.2727	0.4	96	11
aardvark	1.0	0.2	0.3333	88	10
waterbuck	0.6667	0.2222	0.3333	78	9
aardwolf	1.0	0.1111	0.2	73	9
leopard	0.0	0.0	0.0	48	5
bushbuck	0.0	0.0	0.0	34	4
vulture	0.0	0.0	0.0	32	4
duiker	0.0	0.0	0.0	29	3
hyenaStriped	0.0	0.0	0.0	29	3
cattle	0.0	0.0	0.0	27	3
reptiles	0.0	0.0	0.0	17	2
rhinoceros	1.0	1.0	1.0	15	1
steenbok	0.0	0.0	0.0	15	1
honeyBadger	0.0	0.0	0.0	14	1
caracal	0.0	0.0	0.0	14	1
civet	0.0	0.0	0.0	11	1
wildcat	0.0	0.0	0.0	8	1
genet	0.0	0.0	0.0	8	1
zorilla	0.0	0.0	0.0	5	1
rodents	0.0	0.0	0.0	1	1
Macro	**0.5055**	**0.3558**	**0.3944**	**155254**	**19377**

Table 4: Baseline Model - Per Species Statistics.

Species	Precision	Recall	F1-score	Count (Train)	Count (Test)
wildebeest	0.8438	0.9666	0.901	48377	6047
zebra	0.9173	0.9143	0.9158	36480	4560
gazelleThomsons	0.8813	0.9333	0.9066	30368	3795
giraffe	0.9112	0.7857	0.8438	5000	392
guineaFowl	0.8506	0.6931	0.7638	5000	189
hartebeest	0.77	0.522	0.6222	5000	295
impala	0.7055	0.7055	0.7055	5000	343
otherBird	0.6058	0.4242	0.499	5000	297
buffalo	0.765	0.6776	0.7186	5000	490
gazelleGrants	0.5312	0.381	0.4437	5000	357
warthog	0.7205	0.603	0.6565	5000	466
elephant	0.8893	0.7576	0.8182	5000	594
hyenaSpotted	0.6368	0.5644	0.5984	5000	264
lionFemale	0.7368	0.56	0.6364	4828	150
topi	0.6552	0.38	0.481	4816	150
secretaryBird	0.9565	0.7333	0.8302	4410	90
eland	0.8776	0.4778	0.6187	4338	90
baboon	0.8481	0.536	0.6569	4012	125
ostrich	0.6296	0.2881	0.3953	3824	59
cheetah	0.6452	0.3448	0.4494	3760	58
reedbuck	0.8889	0.5614	0.6882	3696	57
hippopotamus	0.8592	0.8026	0.8299	3648	76
dikDik	0.5435	0.6849	0.6061	3540	73
koriBustard	0.7353	0.463	0.5682	3480	54
insectSpider	0.375	0.1915	0.2535	3080	47
jackal	0.4545	0.3191	0.375	3064	47
lionMale	0.6154	0.2222	0.3265	2352	36
serval	0.8889	0.2667	0.4103	1936	30
hare	0.9048	0.7037	0.7917	1760	27
mongoose	1.0	0.1429	0.25	1344	21
batEaredFox	1.0	0.1818	0.3077	960	11
porcupine	0.7143	0.3571	0.4762	960	14
vervetMonkey	0.25	0.0833	0.125	944	12
aardvark	0.3333	0.2	0.25	880	10
waterbuck	0.0	0.0	0.0	780	9
aardwolf	0.5	0.2222	0.3077	730	9
leopard	0.0	0.0	0.0	480	5
bushbuck	0.5	0.25	0.3333	340	4
vulture	0.0	0.0	0.0	320	4
duiker	0.0	0.0	0.0	290	3
hyenaStriped	0.0	0.0	0.0	290	3
cattle	0.0	0.0	0.0	270	3
reptiles	0.0	0.0	0.0	170	2
rhinoceros	0.3333	1.0	0.5	150	1
steenbok	0.0	0.0	0.0	150	1
honeyBadger	0.0	0.0	0.0	140	1
caracal	0.0	0.0	0.0	140	1
civet	0.0	0.0	0.0	110	1
wildcat	0.0	0.0	0.0	80	1
genet	0.0	0.0	0.0	80	1
zorilla	0.0	0.0	0.0	50	1
rodents	0.0	0.0	0.0	10	1
Macro	**0.5053**	**0.3635**	**0.4012**	**231437**	**19377**

Table 5: ROS Model (two-phase training) - Per Species Statistics.

Species	Precision	Recall	F1-score	Count (Train)	Count (Test)
wildebeest	0.8099	0.9558	0.8768	15000	6047
gazelleThomsons	0.91	0.9086	0.9093	15000	3795
zebra	0.8951	0.9154	0.9051	15000	4560
elephant	0.8598	0.7643	0.8093	4761	594
buffalo	0.8478	0.6367	0.7273	3920	490
warthog	0.7104	0.6159	0.6598	3730	466
giraffe	0.9144	0.6811	0.7807	3145	392
gazelleGrants	0.6383	0.3361	0.4404	2857	357
impala	0.727	0.691	0.7085	2746	343
otherBird	0.6703	0.4108	0.5094	2380	297
hartebeest	0.586	0.5661	0.5759	2368	295
hyenaSpotted	0.6622	0.5644	0.6094	2119	264
guineaFowl	0.8035	0.7354	0.768	1516	189
lionFemale	0.7265	0.5667	0.6367	1207	150
topi	0.7528	0.4467	0.5607	1204	150
baboon	0.8701	0.536	0.6634	1003	125
secretaryBird	0.9286	0.7222	0.8125	735	90
eland	0.9286	0.4333	0.5909	723	90
hippopotamus	0.8784	0.8553	0.8667	608	76
dikDik	0.6825	0.589	0.6324	590	73
ostrich	0.6364	0.2373	0.3457	478	59
cheetah	0.8333	0.3448	0.4878	470	58
reedbuck	0.85	0.5965	0.701	462	57
koriBustard	0.5472	0.537	0.5421	435	54
insectSpider	0.3143	0.234	0.2683	385	47
jackal	0.5	0.234	0.3188	383	47
lionMale	0.625	0.2778	0.3846	294	36
serval	0.6923	0.3	0.4186	242	30
hare	0.9048	0.7037	0.7917	220	27
mongoose	0.4286	0.2857	0.3429	168	21
porcupine	0.8571	0.4286	0.5714	120	14
vervetMonkey	0.3333	0.1667	0.2222	118	12
batEaredFox	0.6667	0.1818	0.2857	96	11
aardvark	0.6667	0.2	0.3077	88	10
waterbuck	0.0	0.0	0.0	78	9
aardwolf	1.0	0.1111	0.2	73	9
leopard	1.0	0.2	0.3333	48	5
bushbuck	0.0	0.0	0.0	34	4
vulture	0.0	0.0	0.0	32	4
duiker	0.0	0.0	0.0	29	3
hyenaStriped	0.0	0.0	0.0	29	3
cattle	0.0	0.0	0.0	27	3
reptiles	0.0	0.0	0.0	17	2
rhinoceros	1.0	1.0	1.0	15	1
steenbok	0.0	0.0	0.0	15	1
honeyBadger	0.0	0.0	0.0	14	1
caracal	0.0	0.0	0.0	14	1
civet	0.0	0.0	0.0	11	1
wildcat	0.0	0.0	0.0	8	1
genet	0.0	0.0	0.0	8	1
zorilla	0.0	0.0	0.0	5	1
rodents	0.0	0.0	0.0	1	1
Macro	**0.5319**	**0.3648**	**0.4171**	**85029**	**19377**

Table 6: RUS Model (two-phase training) - Per Species Statistics.

Second, the results that we reported at the start of this chapter need to be interpreted with care. In this paper, we aimed to treat each class equally. We did this by weighing our class-specific performance metrics by class rather than by sample and by not excluding any classes, regardless of their size. As a result, we obtained performance metrics that were more informative than the overall accuracy, which is generally biased towards the majority class. However, given that there were many minority classes with very few images in the training and the test set, high increases and decreases in class-specific performances were sometimes observed, though they only represent predictions on a few images. The performance for these extremely small classes mostly remained unchanged over the different models, meaning that their influence on the results is limited. Nevertheless, these classes do contribute to the values for the macro performance metrics and drive the differences of these values between the models slightly towards zero. Therefore, both the macro performance metrics as well as the class-specific performance metrics need to be interpreted cautiously.

Third, the above explained limitation could be partly mitigated by training the different models several times, with a different random initialisation of the weights. This would allow us to report average performances for the different models, which would make our results more robust. However, the large size of the dataset implied long training times and prevented us from pursuing this preferred strategy.