System for analyzing data from camera traps based on a two-stage neural network approach for operational remote monitoring

Vladislav Efremov*, *Andrew* Leus, *Dmitry* Gavrilov, *Daniil* Mangazeev, *Victor* Zuev, *Alexandra* Radysh, *Ivan* Kholodnyak, *Nikita* Vodichev, and *Masoud* Vahid Dastgerdi

Moscow Institute of Physics and Technology, Dolgoprudny, Moscow Region, 141701, Russian Federation

Abstract. Camera traps are a widely used tool for monitoring wildlife with minimal human intervention. Their number can amount to several hundred, and the accumulated volume can reach several terabytes. Often, photos and videos contain empty frames that are created by accidental triggering of camera trap detectors, such as by wind. The staff of nature reserves must process the images manually and sort them by animal species. In our study we propose to consider a technology for analysing data from camera traps using a two-stage neural network processing. The task of the first stage was to separate empty images from non-empty images. To do this, using a comparative analysis, we identified the most optimal detector model from the YOLO series. The task of the second stage was to classify the objects found by the detector. For this purpose, a comparative analysis of the architectures of classifiers from the ResNet series was carried out. Based on the selected algorithms, a two-stage system for processing data from camera traps was created in the form of a graphical interface with the ability to work on any operating system. The software will significantly reduce the processing time of data from camera traps and simplify environmental analysis.

1 Introduction

Camera traps are increasingly used as a tool for non-invasive monitoring in specially protected natural areas (SPNA) [1]. The volumes of data that reserves are forced to store can amount to several terabytes, and not all photographs from this volume of data are informative. Due to the operation of motion sensors, the camera trap can react not only to the target object, but also to weather conditions, such as wind, rain or blizzard, which generates many empty images. The number of camera traps per reserve can amount to several hundred, and today in Russia there are about 300 specially protected natural areas where they are located.

Some protected areas in Russia use standard graphical programs for processing data from camera traps, allowing them to sort animals by species and create registrations. But often

* Corresponding author: efremov.va@phystech.edu

reserve workers are forced to do this manually without using automated data processing tools, which wastes a huge amount of time and human resources.

One way to overcome the problem of manually processing camera trap data is to use deep convolutional neural networks. There are many works where attempts have been successfully made to automatically process data from camera traps, for example in [2], where the authors used convolutional neural networks to recognize animal species on the Seregenti [3] and Panama [4] data sets. In [5], the authors raised the problem of generalizing neural network models to images from new locations. It has been shown that the quality of models degrades greatly in images from camera traps on which the model was not trained. The main disadvantage of the above-mentioned works is the lack of a ready-made and convenient tool for users to automatically find and recognize objects using camera trap materials.

Considering the above-mentioned shortcomings, difficulties and limitations faced by scientists and engineers in the task of processing data from camera traps, we set the following goals in our work:

- Conduct a comparative analysis of modern detectors from the YOLO series in the task of recognizing objects in images
- Select the optimal classifier capable of recognizing detected objects by the detector with high accuracy
- Based on optimally selected algorithms, develop a two-stage system for processing data from camera traps in the form of software.

2 Materials and methods

2.1 Data collection

Data for training detection and classification algorithms were collected from 51 nature reserves located in the Russian Federation. The initial sample has a strong imbalance in classes, is not structured and is heavily cluttered, which is expressed in the form of duplication of some images. As a result, 1 million images and 65 thousand videos from camera traps were collected, and the data was obtained from different climatic zones, with different backgrounds, weather conditions, lighting conditions, time of day, and etc.

2.2 Data processing

In order for algorithms to generalize well to find and classify objects in images, it is necessary to prepare the training data accordingly. In [6, 7], during data labeling for the classification task, the object class was assigned to a sequence of images, rather than to an individual frame. With this approach, there is a possibility of assigning an animal tag to an empty image. In our work, we label each sequence image generated by a camera trap. Having processed the entire data array, 231k labeled images were obtained for the detection task and 416k images for the classification task. To speed up the data labeling process, we used the old model weights of the detector and classificator [8] trained mainly on data from the Central Forest Reserve of Russia. Images for which it was impossible to make an unambiguous decision when creating markup were removed from the sample. Examples of such data are shown in Figure 1.

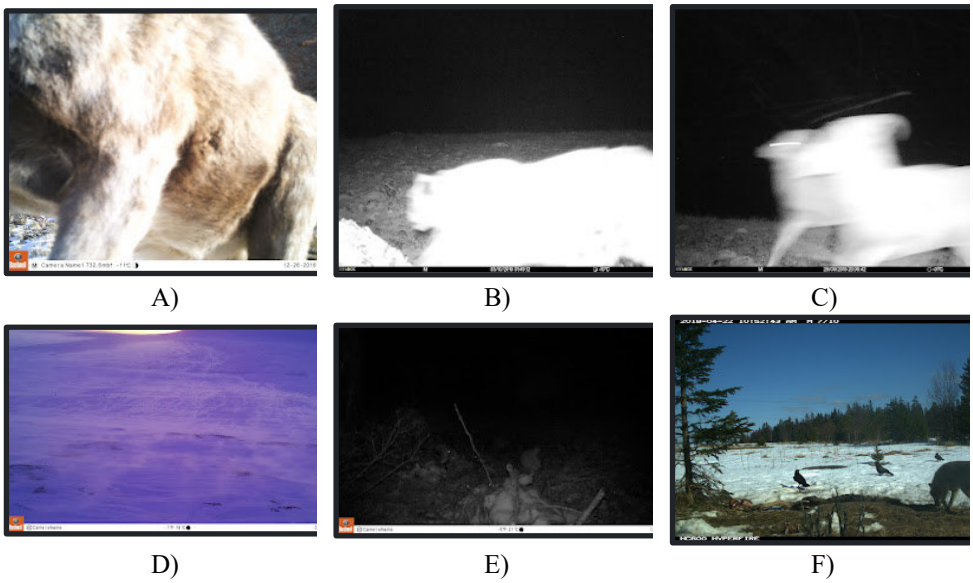

Fig. 1. A) The animal is close to the camera trap, B) Illuminated animal, C) Blurred image, D) Empty image, E) Animal in the dark, F) Several animal species.

The resulting data distribution for the classification task is presented in Figure 2:

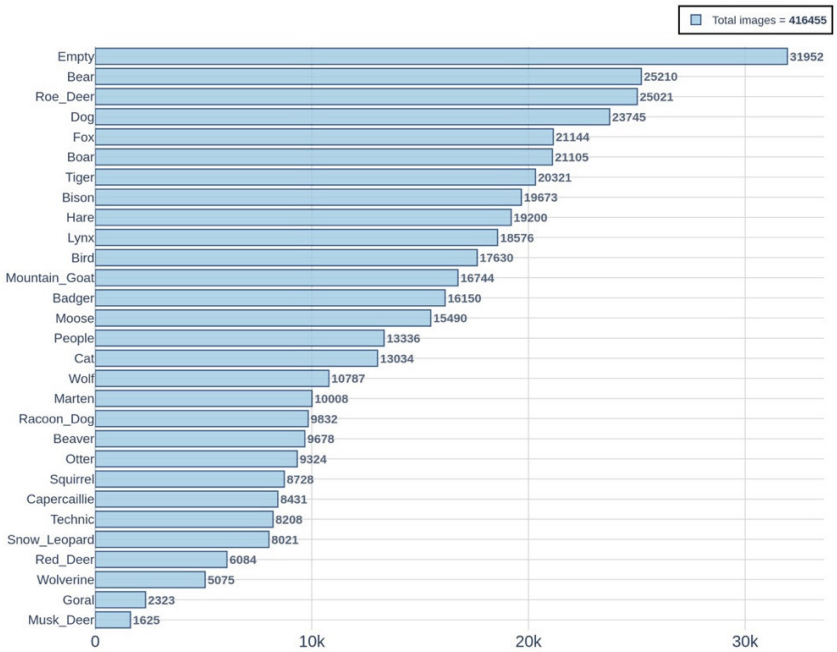

Fig 2. Distribution of animal species for the classification task.

2.3 Detection and classification task

The key task in our study is to train detectors to separate the image background from the animal. The deep neural networks were the algorithms from the YOLO series – YOLOv5 [9], YOLOR [10], YOLOX [11]. The validity of the selected series of algorithms is determined by the fact that they have the best compromise between speed and quality of work.

The comparative analysis of classifiers involved neural network architectures from the ResNet series: ResNet [12], SeResNet [13], ResNeSt [14], ReXNet [15]. The choice is due to the fact that the ResNet model showed one of the best results in the task of recognizing animal species in [2], [7], [16]. We use the transfer learning technique to additionally train model weights on our data. The models were pre-trained on the ImageNet dataset. The models were trained using stochastic gradient descent (SGD) for 35 epochs with a learning coefficient of 0.0003, moment 0.9, batch size 16, EMA weight smoothing. To expand the number of images in the training set, we used data augmentation techniques: horizontal flip, random rotations within 20 degrees, crop. The classifiers were trained on one GeForse RTX 3080 10 GB.

2.4 Metrics in detection and classification tasks

To assess the quality of detection methods, we used the IoU metric (Table 1, A), which allows us to calculate the degree of overlap between the true and predicted bounding boxes, and the average precision metric (Table 1, B), which can be defined as the area under the Precision-Recall curve.

To assess the quality of the classifiers' work, metrics such as Precision, Recall, F1 were used (Table 1, C, D, E). We do not use the accuracy metric because the data set has a strong class imbalance.

Table 1. Metrics for determining the quality of detection (A, B) and classification (C, D, E) algorithms.

$\mathrm{IoU}(A,B) = \frac{A \cap B}{A \cup B}$	$\mathrm{AP} = \frac{1}{n}\sum_{i=1}^{n} p(\tau_i)$	
A)	B)	
$\mathrm{Precision} = \frac{TP}{TP + FP}$	$\mathrm{Recall} = \frac{TP}{TP + FN}$	$F_1 = \frac{2 \cdot \mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$
C)	D)	E)

3 Results

3.1 Results of detection algorithms

We trained YOLOv5-M6/L6/X and YOLOR-W6 at a resolution of 1280, and YOLOv5-X and YOLOX-X were trained at a resolution of 640. Table 2 shows the quality metrics scores on the validation and test datasets.

Table 2. Evaluation of detector quality metrics on validation and test datasets. FPS measured with batch size 1 on RTX 3080.

Model	**Params, M**	**Train resolution**	**Test resolution**	AP_{50}^{val}	$AP_{50:95}^{val}$	AP_{50}^{test}	$AP_{50:95}^{test}$	**FPS**	**Train-time (hours)**
YOLOv 5-M6	35.2	1280	1280	97	82.8	98.4	85.1	42	**38**
			640	96.3	80.4	98.1	83	**71**	
YOLOv 5-L6	76.1	1280	1280	**97.1**	**83.1**	98.5	**85.5**	35	74
			640	96.7	81.2	98.3	83.5	58	
YOLO R-W6	79.2	1280	1280	96.7	80.1	98.1	82.6	33	107
			640	96.1	77.7	97.6	80.3	44	
YOLOv 5-X	86	1280	1280	95.9	76.1	96.9	76.9	27	123
			640	96.4	78.1	97.7	79.2	64	
		640	1280	92.3	69.4	93.8	70.7	27	39
			640	97	82.7	**98.6**	85	64	
YOLO X-X	99.1	640	640	96	78.3	97.5	80.6	35	85

The results show that in the validation sample, the YOLOv5-L6 architecture performed best with AP_{50}^{val} 97.1%, while in the test sample YOLOv5-X trained at 640 resolution performed better than YOLOv5-L6 by 0.1% on the metric AP_{50}^{test}. YOLOv5-L6 was chosen as the final architecture because it has the best compromise between quality and speed among the detectors trained at 1280 resolution. Additionally, detectors trained at 1280 are more likely to find small animals than detectors trained at 640.

We did not consider the larger architectures like YOLOv5-X6, YOLOR-E6, YOLOR-D6 as training such models takes a lot of time on three RTX 3080. In addition, most reserves in Russia have limited computing resources, so inference on heavier models will take more time to process photos and videos.

In the next step, the performance of the YOLOv5-L6 detector was evaluated on a public dataset with animals (https://doi.org/10.5281/zenodo.7215780), where it was trained for 10 epochs with the same parameters. The pre-trained model weights were taken from the previous experiment. The results on the validation and test samples for the metric AP_{50} were 99.4% and 99.5%, respectively.

3.2 Results of classification algorithms

The final architecture is chosen by the F1 metric, as the accuracy metric is unstable to unbalanced data. Table 3 shows the results of the algorithms for the Precision, Recall, F1 metrics on the validation and test datasets.

Table 3. Evaluation of classifier quality metrics using validation and test data. Image size - resolution of the images on which the neural network was trained.

Model	Params, M	Image size	P^{val}	R^{val}	F_1^{val}	P^{test}	R^{test}	F_1^{test}	Train-time (hours)
SeResNet-152	66.8	256x256	98.521	97.635	98.040	98.515	98.107	98.301	44
ResNeSt-101	48.3	256x256	98.307	98.304	98.304	98.182	98.533	98.339	42
ReXNet-100	48	224x224	98.014	97.347	97.656	98.076	97.366	97.694	15
ResNet-101	44.6	256x256	98.262	96.876	97.466	98.032	96.939	97.408	33

The results on the test data show that ResNeSt-101 architecture performs best in animal species recognition task with F1 metric of 98.304% and 98.339% on validation and test data, respectively. The basic ResNet-101 has the lowest F1 score of 97.466% and 97.408% on validation and test data, respectively.

Then we evaluated the performance of the ResNeSt-101 network on an open dataset (https://doi.org/10.5281/zenodo.7215780). The classificator was fed with cropped photos by the bounding boxes. The model weights were taken from the previous experiment and refined over 10 epochs. The rest of the parameters were taken from the previous experiment. The performance of the architecture on the validation and test datasets on the F1 metric was 99.6% and 99.5%, respectively.

3.3 Graphical interface for users

We combined the YOLOv5-L6 algorithm and ResNeSt-101 and made a two-stage system for processing photo and video from camera traps. This is implemented in the form of a graphical interface written using the cross-platform Avalonia UI framework. The user has the possibility to specify the path to the directory with photos and videos he/she wants to process and select the weights of the pre-trained detector and classificator. The above are the minimum requirements to process any directory. For more advanced users it is possible to make fine adjustments to the inference, changing e.g., image size, batch size, detector threshold etc. (Figure 3).

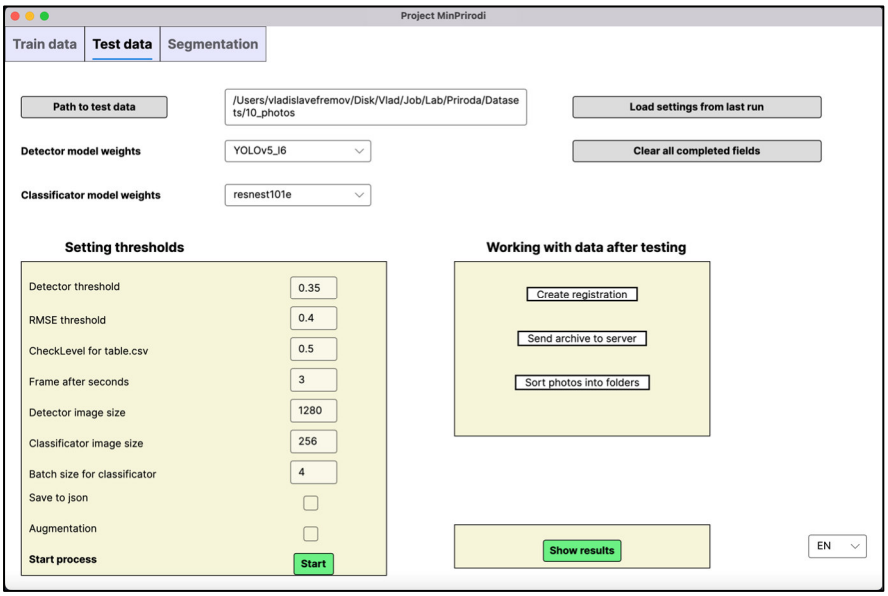

Fig. 3. User window view of a graphical application for processing photo and video from camera traps using a two-stage neural network approach.

Based on the results of the previous section, it can be seen that the training time of a classifier on one GPU is 3-9 times less than the training time of a detector on 3 GPUs. In this regard, retraining the detector for each reserve separately requires large computational resources, but will not provide a significant increase in the quality of recognition. Therefore, it was decided to add functionality to the software that allows you to retrain the classifier for the species diversity of a particular reserve, but at the same time use a single universal detector.

4 Discussion

The work demonstrates a two-stage approach to processing data from camera traps based on the optimally selected YOLOv5-L6 detector and ReNeSt-101 classifier, which showed the best results in finding and classifying animal species. Primary image processing by the detector allows you to separate the image background from the object, which more effectively affects the quality of data classification.

Most studies focus on the generalization of neural network models for the maximum diversity of animals [5, 6, 17, 18]. This paper proposes a tool that can be used to retrain a classifier for the species diversity of a particular reserve. This approach allows to reduce the number of errors in the prediction of animal species and reduces the time for manual correction of neural network predictions for reserve workers. At the same time, due to the significant difference in training time between the classifier and the detector, it is proposed to use a single universal detector for detecting animals, which does not need to be retrained.

It is worth noting that nature reserves have very limited computing resources and sometimes computers are not equipped with powerful graphics cards. Therefore, it is important that the system processes the image at a speed higher than what a reserve worker does manually. We conducted experiments on processing 1000 images and 100 video files on computers with different configurations. The experimental results are shown in Table 4.

Table 4. Estimation of the speed of a graphical application installed on computers with different configurations.

CPU	**RAM**	**Graphics card**	**Image size**	**Data processing time**	
				1000 photos	***100 videos***
M1	**Mac OS**	**-**	**640**	**53 minutes**	**46 minutes**
Intel Core i7-6700HQ CPU 2.6GHz	Linux (Fedora)	GTX 960M (2 ГБ) (notebook)	1280	26 minutes 34 seconds	24 minutes 24 seconds
Intel Core i7-8750H CPU 2.20GHz	Windows 10	GTX 1050 Ti (4 ГБ) (notebook)	1280	8 minutes 44 seconds	8 minutes 15 seconds
Intel Core i7-10875H CPU 2.30GHz		RTX 2070 (8 ГБ) (notebook)	1280	2 minutes 27 seconds	2 minutes 50 seconds
Intel Core i7-10700F CPU 2.90GHz	Linux (Ubuntu 21.10)	RTX 3080 (10 ГБ) (desktop)	1280	1 minute 30 seconds	1 minute 53 seconds

Table 4 shows that with a modern GPU (RTX 3080) the image processing speed reaches 11 images per second, well beyond human capability. Whereas with an older configuration (GTX 960M) the processing speed is 1 image in 3 seconds, which is roughly comparable to actual human image processing speed.

Often, work on automatic processing of data from camera traps consists of program code on github with console launch of programs. Taking this shortcoming into account, we created a graphical application for users with the ability to install it on any operating system. Using the developed interface, users can automatically process photo and video materials from camera traps.

References

1. A.F. O'Connell, J.D. Nichols, K.U. Karanth, *Camera traps in animal ecology: Methods and analyses* (Berlin, Germany: Springer Science & Business Media, 2011), p. 279. https://doi.org/10.1007/978-4-431-99495-4
2. A. Gomez-Villa, A. Salazar, F. Vargas, Ecological Informatics **41**, 24-32 (2017). https://doi.org/10.1016/j.ecoinf.2017.07.004
3. A. Swanson, M. Kosmala, C.J. Lintott, R. Simpson, A.M. Smith, C. Packer, Scientific Data **2**, 150026 (2015)

4. G. Chen, T.X. Han, Z. He, R.W. Kays, T.D. Forrester, *Deep convolutional neural network based species recognition for wild animal monitoring,* 2014 IEEE International Conference on Image Processing (ICIP), 858-862 (2014)
5. S. Beery, G. Van Horn, P. Perona. (2018). Recognition in Terra Incognita. Computer Vision – ECCV 2018. Lecture Notes in Computer Science, vol 11220. (Springer, Cham, 2018). https://doi.org/10.1007/978-3-030-01270-0_28
6. M.S. Norouzzadeh, A. Nguyen, M. Kosmala, A. Swanson, M.S. Palmer, C. Packer, J. Clune, Proceedings of the National Academy of Sciences of the United States of America **115(25)**, E5716–E5725 (2018). https://doi.org/10.1073/pnas.1719367115
7. M. Willi, R.T. Pitman, A.W. Cardoso, et al., Methods in Ecology and Evolution **10(1)**, 80-91 (2019). https://doi.org/10.1111/2041-210X.13099
8. A.V. Leus, V.A. Efremov, Proceedings of the Mordovia State Nature Reserve **28**, 121-129 (2021).
9. J. Glenn, YOLOv5 release v6.1. (2021). https://github.com/ultralytics/yolov5/releases/tag/v6.1
10. C. Wang, I. Yeh, H.M. Liao, You Only Learn One Representation: Unified Network for Multiple Tasks (2021). https://doi.org/10.48550/arXiv.2105.04206
11. Z. Ge, S. Liu, F. Wang, Z. Li, J. Sun, YOLOX: Exceeding YOLO Series in 2021 (2021). https://doi.org/10.48550/arXiv.2107.08430
12. K. He, X. Zhang, S. Ren, J. Sun, *Deep residual learning for image recognition,* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770–778 (2016). https://doi.org/10.1109/CVPR.2016.90
13. J. Hu, L. Shen, S. Albanie, G. Sun, E. Wu, IEEE Transactions on Pattern Analysis and Machine Intelligence **42**(8), 2011-2023 (2020). https://doi.org/10.1109/TPAMI.2019.2913372
14. H. Zhang, C. Wu, Z. Zhang, Y. Zhu., Z. Zhang, H. Lin, Y. Sun, T. He, J. Mueller, R. Manmatha, M. Li, A. Smola, ResNeSt: Split-Attention Networks (2020). https://doi.org/10.48550/arXiv.2004.08955
15. D. Han, S. Yun, B. Heo, Y.J. Yoo, ReXNet: Diminishing Representational Bottleneck on Convolutional Neural Network (2020). https://doi.org/10.48550/arXiv.2007.00992
16. 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, B. Teton, J.C. Beasley, P.E. Schlichting, R.K. Boughton, B. Wight, E.S. Newkirk, J.S. Ivan, E.A. Odell, R.K. Brook, R.S. Mille, Methods in Ecology and Evolution **10(4)**, 585-590 (2018). https://doi.org/10.1111/2041-210X.13120
17. C. Zhu, T. H. Li, G. Li, *Towards Automatic Wild Animal Detection in Low Quality Camera-Trap Images Using Two-Channeled Perceiving Residual Pyramid Networks*, 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), 2860-2864 (2017). https://doi.org/10.1109/iccvw.2017.33
18. H. Yousif, J. Yuan, R. Kays, Z. He. (2017). Fast human-animal detection from highly cluttered camera-trap images using joint background modeling and deep learning classification. 2017 IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1-4, https://doi.org/10.1109/ISCAS.2017.8050762