Reevaluating Automated Wildlife Species Detection: A Reproducibility Study on a Custom Image Dataset

Preprint, compiled September 13, 2025

Tobias Abraham Haider [1, 2*]

[1]Vienna University of Technology
[2]University of Veterinary Medicine Vienna

Abstract

This experiment reproduces the results of the paper Automated detection of European wild mammal species in camera trap images with an existing and pre-trained computer vision model [1], which tests the pretrained Google Inception-ResNet-v2 model for animal species identification. We describe the required software, image loading processes, and model outputs. Furthermore, we calculate the prediction accuracies for each present species and the whole dataset and compare them to the metrics from the original paper. The observed total prediction accuracy of 62% comes close to the reported 71% by Carl et al. The large difference in per-class accuracy, ranging from 0% to 100%, can also be observed in our experiment. Like Carl et al., we recommend the use of the pretrained Inception-ResNet-v2 model for simple animal species identification tasks, emphasizing the need to refit the model to the species relevant for the specific use case if high prediction accuracies and consistency are desired.

Keywords machine learning, reproducibility, camera tramp, pre-trained model, animal species classification, computer vision, neural networks, cnn, resnet, tensorflow, wildlife monitoring

1 Introduction

While biodiversity is decreasing at a rapid pace, the rise of specific species, be they invasive or predatory, concerns societies around the world. As a consequence, researchers and conservationists are interested in continuously monitoring wildlife populations in terms of their geographical distribution, size, and behavior. Researchers successfully deploy camera traps that can take photographs of passing animals without disturbing them [2]. The photos are typically manually collected from the traps and annotated with the name of the species present in the image [3]. This experiment tests one popular software for eliminating the need for manual annotation of images: deep convolutional neural networks.

2 Experiment Setup

We decide to reimplement the Python code for the experiment from scratch because all the necessary components (data [4], model [5], and metrics [6]) can be taken from stable public sources. To maximize the readability and reproducibility of the experiment, a minimal setup was chosen, defining all necessary code, data, and requirements in one GitHub project. State-of-the-art Python packages are chosen, installed, and imported. The exact versions are shown in Table 1. The Jupyter notebook is run locally on a Thinkpad T14 with an AMD Ryzen 5 PRO 5650U processor, 16 GB of memory, and Linux Mint 22.1 installed. No GPU was used, but it can be expected that the results would not change if one were used.

```
!python3.12 -m pip install -r requirements.txt
```

Table 1: Runtime dependencies

package	version
pathlib	1.0.1
Pillow	11.3.0
numpy	2.1.3
pandas	2.3.1
tensorflow	2.19.0
scikit-learn	1.7.1

3 Model

After setting up the Python runtime and importing the packages, the Inception-ResNet-v2 model is downloaded from the TensorFlow repository [7] and directly fit to the ImageNet dataset [8]. This eliminates all model design and training work.

```
model = InceptionResNetV2(weights="imagenet")
```

4 Data

Carl et al. provide the source for their wildlife images for their dataset [9]. This source is no longer available, requiring us to run the experiment on a different dataset. To test the generalizability of the model, we take a larger public dataset containing images of 90 different species [4]. To mimic the original experiment setup, only 10 samples are used for each species, resulting in a total test sample size of 900 images.

4.1 Data Preprocessing

We load the images respecting all three color channels (RGB), resize them to 299 by 299 pixels, and convert them into a 1-dimensional vector. The color intensities are scaled to be floating-point numbers from 0 to 1. This is the minimal preprocessing required to fit the required input size of the neural network.

```
def load_image(path, target_size):
    image = Image.open(path).convert("RGB")
    image = image.resize(target_size)
    return np.array(image) / 255.0
```

Then we construct the testing dataset by stacking all normalized image vectors and using the folder names as the labels.

```
animal_images = [load_image(p, input_shape)
                 for p in wildlife_image_paths]
animal_species = [p.parent.name
                  for p in wildlife_image_paths]

X_test = np.stack(animal_images, axis=0)
y_true = animal_species
```

5 TEST

The model yields a probability for each of the 1000 classes. The classes represent 1000 different classes taken from the ImageNet database. For this experiment, we use the output from the top neuron of the final softmax layer and compare its label to the true label.

```
y_pred = model.predict(X_test)
y_pred = [pred[0][1] # take output label
          for pred
          in decode_predictions(y_pred, top=1)]
```

When looking at the results, it becomes apparent that the model yields usable results. Almost all inference outputs are animal species somehow related to the one present in the image. This already shows that the InceptionResNetV2 is generalizable to some extent.

5.1 Label Mapping

The main issue with this experiment is the set of classes known to the model, which do not match the dataset used for testing. This is not unique to this dataset, but it is very likely to happen in any kind of realistic setup. We manually define a mapping table to relate the model output label to the labels from the dataset.

Table 2: Subset of Inception-ResNet-v2 raw predictions

y_pred	y_true
gazelle	antelope
badger	badger
hummingbird	bat
brown_bear	bear
bee	bee
honeycomb	beetle
bison	bison
wild_boar	boar
ringlet	butterfly
Egyptian_cat	cat

Table 3: Mapping rules between ImageNet classes and test data classes

ImageNet label	dataset label
gazelle, impala	antelope
American_black_bear, brown_bear	bear
ground_beetle, leaf_beetle, rhinoceros_beetle, dung_beetle	beetle
wild_boar	boar
ringlet, monarch, sulphur_butterfly, lycaenid	butterfly
Egyptian_cat, tabby, Siamese_cat, Persian_cat, lynx	cat
water_buffalo	cow
Dungeness_crab, fiddler_crab, rock_crab, king_crab	crab
magpie, jay	crow
red_deer, elk	deer

The mapping rules are defined manually, following a best-effort approach respecting the Linnean system of taxonomy, and we acknowledge some of its shortcomings:

- Some semantic information is lost as species are sometimes mapped to their families. (e.g., all bear species are mapped to bear)
- The dataset contains images of species that are not directly related to any class from the model. (e.g., all bats, deers)

6 EVALUATION

Carl et al. provide two kinds of performance metrics: overall model accuracy and the accuracy for each species. By grouping the samples by true species, it is straightforward to calculate metrics that are semantically equivalent and therefore allow us to use them for comparison.

```
accuracy = accuracy_score(y_true, y_pred_mapped)
```

Table 4: Prediction accuracy per species and the total accuracy

species	accuracy
bison, bear, boar, crab, elephant, eagle, dog, chimpanzee, cockroach, snake, panda, pelecaniformes, pig, koala, orangutan, ladybug, leopard, lobster, hornbill, jellyfish, hyena, hummingbird, goose, goldfish, fox, fly, sandpiper, zebra, wombat, turtle	1.0
parrot, shark, starfish, squirrel, otter, kangaroo, penguin, coyote, butterfly, flamingo, badger, bee, antelope, hare, gorilla, porcupine, tiger, hamster	0.9
sheep, lizard, lion, cat, dragonfly, wolf	0.8
beetle, hippopotamus, ox, grasshopper	0.7
whale	0.6
duck	0.4
owl, goat, crow	0.3
swan	0.1
caterpillar, bat, dolphin, donkey, cow, deer, mosquito, horse, hedgehog, okapi, moth, mouse, octopus, seal, raccoon, rat, possum, pigeon, oyster, seahorse, rhinoceros, reindeer, squid, sparrow, turkey, woodpecker	0.0
...	...
TOTAL	0.62

7 SUMMARY

TODO: Carl et al. did a good job and their results are valid. It is important to emphasize the fact that the model can only predict a limited number of species. This makes the model almost certainly unsuitable for use in real use cases. The neural network architecture is powerful and very accurate in predicting classes that it was trained on.

8 FUTURE WORK

TODO: Find paper that demonstrates transfer lerning using the inceptionresnet model and explain how great accuracy can be achived also for other species. Also mention that there are still few projects that deploy camera traps with such models and really have all of this automated.

REFERENCES

[1] Christin Carl, Fiona Schönfeld, Ingolf Profft, Alisa Klamm, and Dirk Landgraf. Automated detection of European wild mammal species in camera trap images with an existing and pre-trained computer vision model. *European Journal of Wildlife Research*, 66(4), 7 2020. ISSN 1439-0574. doi: 10.1007/s10344-020-01404-y. URL http://dx.doi.org/10.1007/s10344-020-01404-y.

[2] Franck Trolliet, Marie-Claude Huynen, Cédric Vermeulen, and Alain Hambuckers. Use of camera traps for wildlife studies. A review. *Biology Agriculture Science Environnement*, 18:446–454, 1 2014.

[3] Dhruv Tulasi, Alys Granados, Prabath Gunawardane, Abhay Kashyap, Zara McDonald, and Sunil Thulasidasan. Smart Camera Traps: Enabling Energy-Efficient Edge-AI for Remote Monitoring of Wildlife. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on AI-Driven Spatio-Temporal Data Analysis for Wildlife Conservation*, GeoWildLife '23, pages 9–16, Hamburg, Germany, 2023. Association for Computing Machinery. ISBN 9798400703553. doi: 10.1145/3615893.3628760. URL https://doi.org/10.1145/3615893.3628760.

[4] Sourav Banerjee. Animal Image Dataset (90 Different Animals). https://www.kaggle.com/datasets/iamsouravbanerjee/animal-image-dataset-90-different-animals, 2024. URL https://www.kaggle.com/datasets/iamsouravbanerjee/animal-image-dataset-90-different-animals. Accessed: 2025-09-03.

[5] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning, 2016. URL https://arxiv.org/abs/1602.07261.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[7] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 6 2009. doi: 10.1109/cvpr.2009.5206848. URL http://dx.doi.org/10.1109/CVPR.2009.5206848.

[9] Nationalparkverwaltung Hainich, FFK Gotha. Schwarzwildforschung im Hainich.

https://www.schwarzwild-hainich.de, 2019. URL https://www.schwarzwild-hainich.de.