**University of Bern**

**CAS Applied Data Science**

**Module 6**

**Elena Beletsioti**

Sophie-Taueber-Str. 6

8050 Zurich

e.beletsioti@gmail.com

**Data Science Project**

**Image Classification in Wildlife Conservation**

**31 August 2023**

# Abstract

Current animal monitoring methods rely on human field workers for data collection, but they are time-consuming, labor-intensive, expensive [1], and can result in biased datasets [2]. Human presence poses risks to wildlife, habitats, and researchers [3]. Additionally, human limitations constrain the scale, resolution, and complexity of data collection. To overcome these challenges, there is a need for alternative, technology-driven monitoring approaches that are efficient, accurate, and safe. By using camera traps [4] and leveraging relevant advances in technology, we can develop innovative approaches that provide valuable insights into wildlife conservation [5]. These approaches have the potential to enhance our ability to assess and monitor animal populations, enabling effective conservation efforts in the face of this crisis.

This Data Science Project Report presents a challenge focused on classifying species appearing in camera trap images collected from Taï National Park. The images are provided by the Wild Chimpanzee Foundation and the Max Planck Institute for Evolutionary Anthropology. Both have provided a substantial dataset comprising camera trap images from various locations within the park, which consists of images featuring seven distinct types of animals: birds, civets, duikers, hogs, leopards, other monkeys, and rodents. Additionally, there are images where no animals are present. The primary objective of this project is to construct an image recognition model capable of accurately predicting whether an image contains one of the aforementioned species.

Here, we utilized and tested two pre-trained Convolutional Network Models, with the aim to automatically identify camera trap images of the aforementioned species. The models deployed show promise, however in terms of accuracy both models did not perform as well as we would have hoped. However, we believe that with some improvements and further iterations the models would perform well in recognizing animals from camera trap images, therefore helping to automatically detect images of animals and classify their species. This predictive model can be the basis for building a robust machine learning model, which can assist researchers in identifying different species for effective wildlife conservation measures.

# Table of Contents

[Abstract 1](#_Toc143681805)

[Table of Contents 2](#_Toc143681806)

[1 Project Objectives 3](#_Toc143681807)

[2 Methods 3](#_Toc143681808)

[2.1 Infrastructure 3](#_Toc143681809)

[2.2 Software, environments, packages and libraries 3](#_Toc143681810)

[2.3 Deep Learning methods for wildlife species recognition 4](#_Toc143681811)

[3 Data 5](#_Toc143681812)

[3.1 Dataset 5](#_Toc143681813)

[3.2 Processing 6](#_Toc143681814)

[3.3 Data Quality 6](#_Toc143681815)

[3.4 Metadata 7](#_Toc143681816)

[4 Exploratory data analysis 7](#_Toc143681817)

[4.1 Exploring the species 7](#_Toc143681818)

[4.2 Distribution of Species 7](#_Toc143681819)

[5 Machine Learning Analysis 8](#_Toc143681820)

[6 Results Discussion 11](#_Toc143681821)

[7 Conclusion and Outlook 13](#_Toc143681822)

[References 15](#_Toc143681823)

# 1 Project Objectives

Camera traps have revolutionized ecological research, enabling the study of wildlife species distribution, activity patterns, and interactions within ecological communities. While camera traps offer a cost-effective means of monitoring numerous species across large spatial and temporal scales, data processing time can hinder survey efficiency. To address this challenge, artificial intelligence (AI), particularly Deep Learning, has garnered significant attention for processing camera trap data. [6]

Deep learning has found widespread application in various tasks related to camera trap data analysis. It has been utilized for removing blank photos (i.e., photos without animals) [7], species identification, individual recognition, and counting of individuals [8]. Deep Learning enables the processing of large volumes of images in short time periods, however it requires substantial amounts of pre-processed data for model training. Also, the performance of Deep Learning approaches may be impacted when applied to new environments, emphasizing the need for adaptation and generalization. [7]

Deep learning techniques, such as convolutional neural networks (CNNs), are trained to utilize specific features within camera trap images for automated object detection (e.g., animals, humans, vehicles) and species classification. By leveraging deep learning algorithms, camera trap data can be processed more efficiently, facilitating the analysis of large datasets and enabling ecologists to extract valuable insights from the collected imagery. [9]

In this data science project, the aim is to utilize machine learning methods for automatic recognition of wildlife, specifically classifying seven species appearing in camera trap images in the Taï National Park, a national park in Côte d'Ivoire, collected by the Wild Chimpanzee Foundation and the Max Planck Institute for Evolutionary Anthropology. The primary objective of this project is to develop an automated image classification model which can identify images containing the seven species, i.e. birds, civets, duikers, hogs, leopards, other monkeys, and rodents, while also identifying those containing no animals (blanks) from camera trap images. This predictive model will assist researchers in identifying and tracking different species for effective conservation measures.

# 2 Methods

## 2.1 Infrastructure

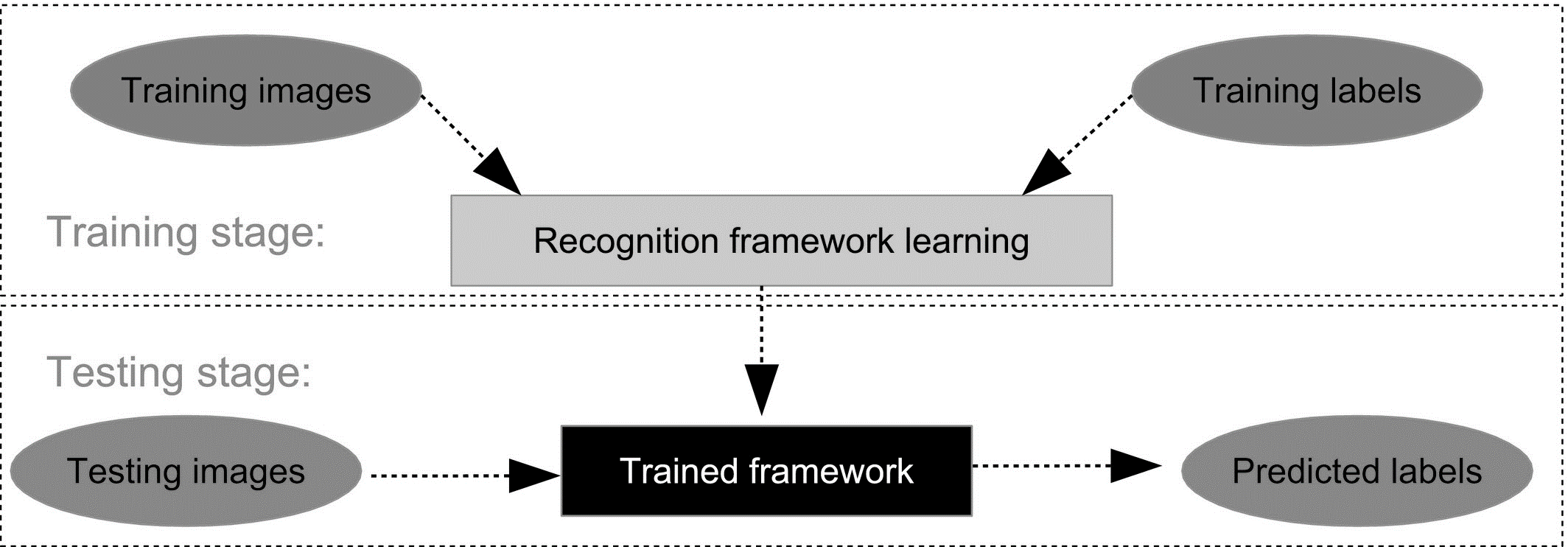
The machine used for the project is an DELL Inspirion 5390 using an Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz, 1800 MHz, 4 Cores, 8 Logical Processors. The operating system is Microsoft Windows 10 Home.

## 2.2 Software, environments, packages and libraries

The Anaconda3-2019.07-Windows-x86\_64 software distribution is used to perform the data analysis. The development environment is Jupyter Notebook 13.0.0. The code is written in Python 3. The following Python libraries were used in this project: pandas, numpy, scipy, matplotlib, seaborn. The following PyTorch libraries were used for machine learning: torch, torchvision, torchvision.models, torch.utils.data.

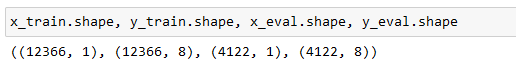
## 2.3 Deep Learning methods for wildlife species recognition

Building an image recognition framework consists of two stages: training and testing (Figure 1). In the training stage, the framework learns parameters from labeled training images, where each image is manually labeled with the corresponding animal shown. In the testing stage, the trained framework receives incoming images as input and generates predictions of the corresponding labels. This process enables automated recognition and classification of animals based on input images.



*Figure 1: The training and testing processes of a recognition framework*

In our project, we split the images into training and evaluation datasets, with a split in 75% for train and 25% for evaluation. Figure 2 illustrates the split.



*Figure 2: Resulting training and evaluation sets are stored in x\_train, x\_eval, y\_train, and y\_eval variables*

In terms of methods, we utilize and test two pre-trained models, which will be described below.

For Model 1, we leverage ResNet50, a recognized network for image classification. The pretrained model generates a 2048-dimensional embedding, linked to two dense layers. Between these layers, we applied a ReLU activation function and a Dropout step. Our model's final layers (model.fc) act as the new "head," transforming embeddings into the needed 8-dimensional output in order to correspond to our 8 species classes, including blank images. Initially, model.fc linked the 2048-dimensional embedding to a 1000-dimensional output, covering ImageNet classes. We adapted it by redefining model.fc for our 8 species, introducing extra layers. A ReLU layer enhanced non-linearity, activating features and reducing noise. Dropout, a common regularization, randomly omitted nodes (10% here) from the previous layer during each training step to prevent overfitting.

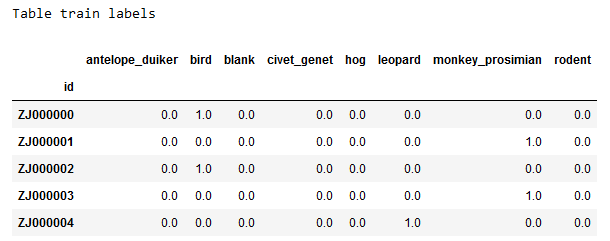
For Model 2, we utilize AlexNet, another CNN-based model, which was originally trained on ImageNet, a vast dataset of 1.2 million labeled images across 1,000 categories. AlexNet consists of 8 layers and we applied the ReLu activation function. AlexNet is supposed to be faster than the very slow ResNet model, this is why we opted for this model to be tested, instead of the VGG-16, which might be also very interested to test on this dataset.

# 3 Data

## 3.1 Dataset

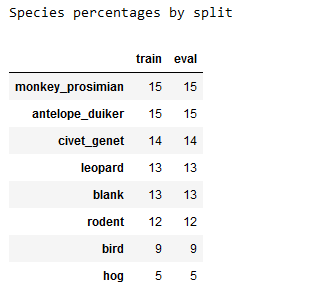
The dataset contains 20.952 camera trap images both taken at day and at night, which were captured at the Tai National Park in Côte d'Ivoire and divided into two folders: Training images, which were assigned to either birds, civets, duikers, hogs, leopards, other monkeys, and rodents. There are also images which contain no animal (blank). Each record in the dataset corresponds to a single image captured by a camera trap. Each record has one .jpg file associated with it in either the train\_features or test\_features directory, depending on which dataset it is a part of. Each image is accompanied by additional information in train\_features.csv and test\_features.csv, which contain the following fields:

* id (string, unique identifier) - a unique identifier for each image
* filepath (string, feature) - image path including its split directory (train or test)
* site (string, feature) - the site in which the image was taken



*Figure 3: Preview of Table train labels*

We randomly selected 75% of the images to be used in the training process, and 25% were put aside for the evaluation. We also validated that our split did result in approximately similar relative distributions of the seven species across both, the train and evaluation sets (Figure 4) by using the stratify parameter.



*Figure 4: Percentage of species in training and evaluation set*

The provided images also have one folder with test images, which contains 4.4.64 images, for which no label/categorization is provided. These are the images the model will aim to predict.

In general, we notice images taken from camera traps are quite challenging even for human eyes, and many images contain animals that are far away, too close, or only partially visible (Figure 5 A–C). In addition, different lighting conditions, shadows, and weather can make the information-extraction task even harder (Figure 5 D). Human-volunteer species and count labels are estimated to be 96% and 90% accurate, respectively, vs. labels provided by experts [10]. We have to see how accurate machine learning is compared to that.

|  |  |  |  |
| --- | --- | --- | --- |
| *A. Partially visible animal* | *B. Animal far away* | *C. Close-up shot of animal* | *D. Image taken at night* |

*Figure 5: Challenges in Image Processing*

## 3.2 Processing

In order to be efficient later in the machine learning analysis, preprocessing of the data is required. We define with \_\_len\_\_ the size of the dataset that needs to be returned, \_\_init\_\_ as well as \_\_getitem\_\_ to support the indexing and define some transformations. In terms of transformation we decide for resizing the images to the size 224x224, as ResNet50 as well as AlexNet was trained on this size [12]; we also convert the PIL images to tensor, as PyTorch recognizes it as an RGB image and therefore convert the input values. Additionally, we normalize the image tensors with the mean and standard deviation of ImageNet images, as the ResNet model was trained with this transformation as well [13].

## 3.3 Data Quality

Because of a glitch while downloading the zipfile with the images into Gdrive, somehow some of the images were not downloaded. This did only became apparent when running the code for training the model (initially in Google Colab). Eberytime I run the code, the FileNotFoundError was prompted. My initial thoughts were that Gdrive was not having enough time and it got disconnected while it was trying to run the code. Only after copying the code to a Jupyter Network and downloading and saving the images again into a local folder, the glitch disappeared and I was able to run the code.



*Figure 6: FileNotFoundError in Google Colab*

## 3.4 Metadata

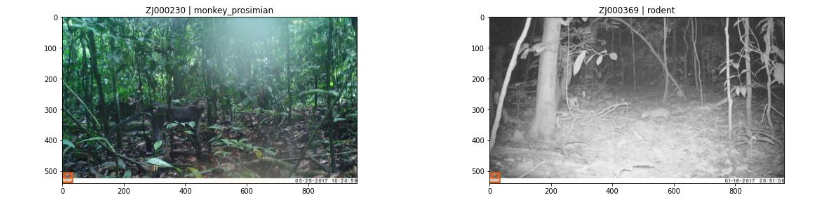
Digital images contain usually contain metadata in standardized Exif format, for example date as well as time and geotags. For each image there is associated metada in the csv file train\_features.csv like image id, filepath and site.

# 4 Exploratory data analysis

## 4.1 Exploring the species

Initially, we plot from each species one random image to display from the dataset, along with its image id and its label, as illustrated in Figure 7.

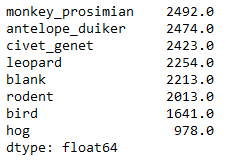
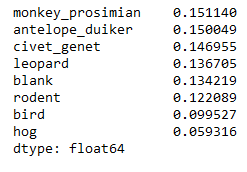




*Figure 7: Plotting of one random image from each species*

## 4.2 Distribution of Species

Additionally, we explore the distribution of species across the training set, first in absolute counts of the species and secondly, looking at the percentages (Figure 8).

*Figure 8: Absolute counts of each species (left), Percentage of each species (right)*

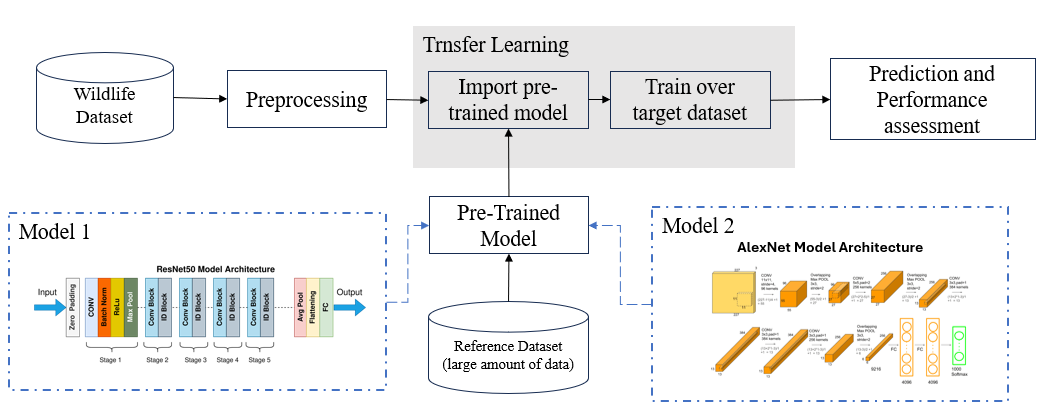
It becomes clear, that only 13% of the captured events were classified as empty of animals, which seems very low as blank images are quite common. A blank image might occur if a sensor is activated due to a temperature difference between a section of vegetation and the background surface temperature [11]. Blank images may also occur, when a camera is triggered by an animal, but the response time is too slow to capture the animal.

Moreover, the dataset is very unbalanced, meaning that some species are much more frequent than others (Monkey 15%, Antelope 15% whereas Bird only 9% and Hog 5%). Such imbalance is problematic for machine learning because they become heavily biased toward classes with more examples. If the model just predicts the frequent classes such as Monkey or Antilope most of the time, it can still get a very high accuracy without investing in learning rare classes.

# 5 Machine Learning Analysis

Studies have proven that Convolutional Neural Networks which have been trained and learned on a large dataset can effectively be transferred to train over a new target dataset. During this process, namely transfer learning, the model has already learned useful features and hence is able to attain a much higher accuracy than any model, which is trained on a smaller dataset  [14].

To identify the animals in the context of classification of wildlife from camera trap data, we trained a CNN, using a pre-trained model, namely ResNet50 (further referred to Model 2). For purposes of comparison, we repeated the training using an AlexNet architecture on the same dataset (further referred to Model 2). The process is illustrated in Figure 9.



*Figure 9: Machine Learning Process with two model architectures*

Comparing two models and assessing their performance is important because each has of the model has advantages and drawbacks. ResNet is highly complex and requires large computational memory, running the code reuired several hours; AlexNet comes with a less complex architecture and seems to require also less computing memory; however, it seems to be more likely to suffer from overfitting.

One of drawbacks for both models is that the performance is highly dependent on the image similarity between the reference dataset (e.g. ImageNet) and target dataset (in this case the Wildlife dataset). The underlying assumption is that two datasets that we used were similar, so we do not expect poor performance due to that fact.

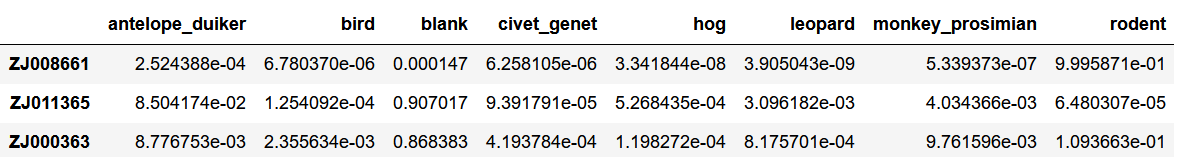
During the training stage, the choice of optimizer was Cross Entropy Loss, as this is a quite common loss function for multi-class image recognition; with momentum of 0.9, and batch size of 128. Since we used, pre-trained models and also running time was very long (the Model 1 took more than five hours to run). We trained Model 1 for 10 epochs with an initial learning rate of 0.001 since weights were randomly initialized. We trained Model 2 also for 10 epochs using an initial learning rate of 0.0001 and experimented with running with more epochs.

Comparing the loss graph for both models, we observe that both loss curves have a tendency to fluctuations in loss over the epochs. Loss of Model 1 reduces at a faster rate when compared to Model 2 and is also more close to decreasing to stability with minor fluctuations (Figure 10). Model 2 the loss is taking much more longer to converge to minima and we can also observe higher tendency to fluctuations with training over more epochs (Figure 11). This might have diverse reasons, and it would be worth experimenting further with the batch size, as the set batch size of 128 might be too small and there is too much noise; as well as with the learning rate, which might cause that we miss the local minimas.

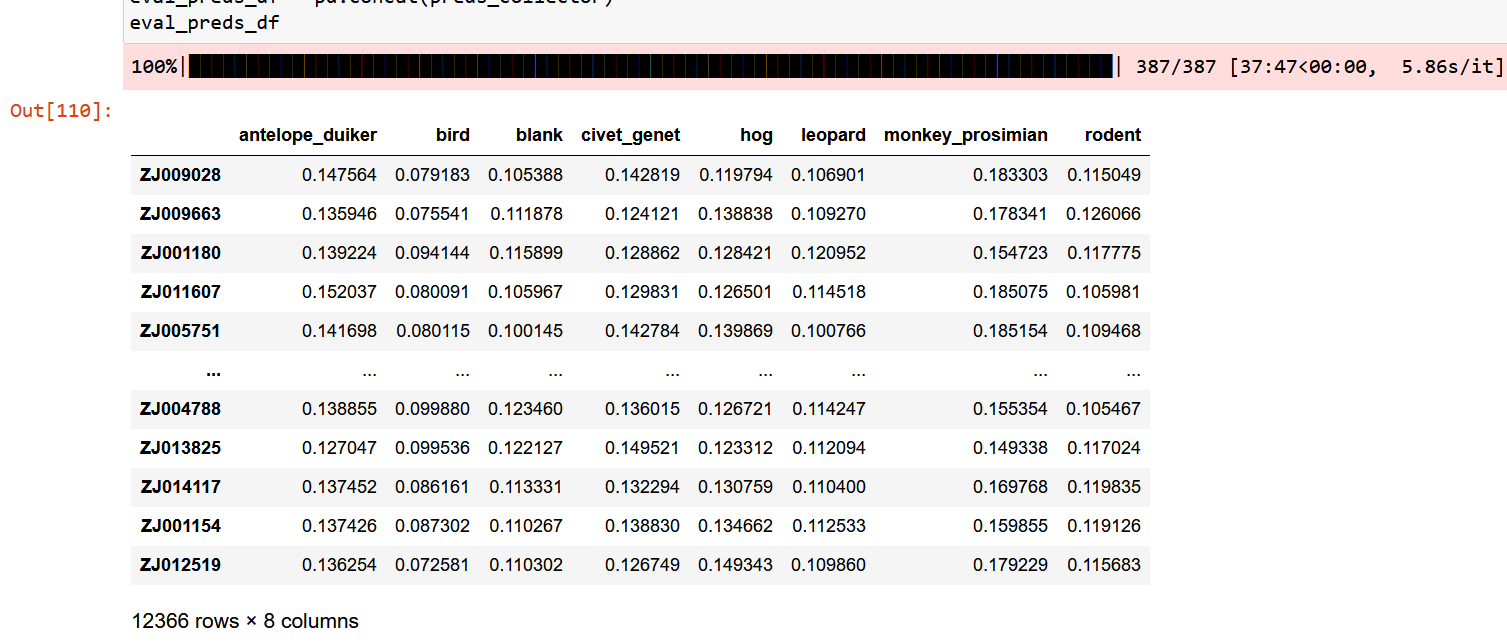
|  |
| --- |
|  |
| *Figure 10: Loss graph for Model 1 for 10 epochs* |

|  |  |
| --- | --- |
|  |  |
| *Figure 11: Loss graph for Model 2 for 10 epochs (left) and for 20 epochs (right)* | |

As a next step, we run the model on the evaluation dataset (which we put aside in the beginning, and for which we do have the true label. The output of the model should be a vector of probabilities of a number between 0 and 1, indicating that the input image belongs to each of the pre-set categories, in this case one of the eight species. Looking at the output data as illustrated in Figure 12 it is obvious that the classification with the model did not work in the manner it should. In previous versions this did not happen (compare with output of Figure 14 where vector was showing some animals).

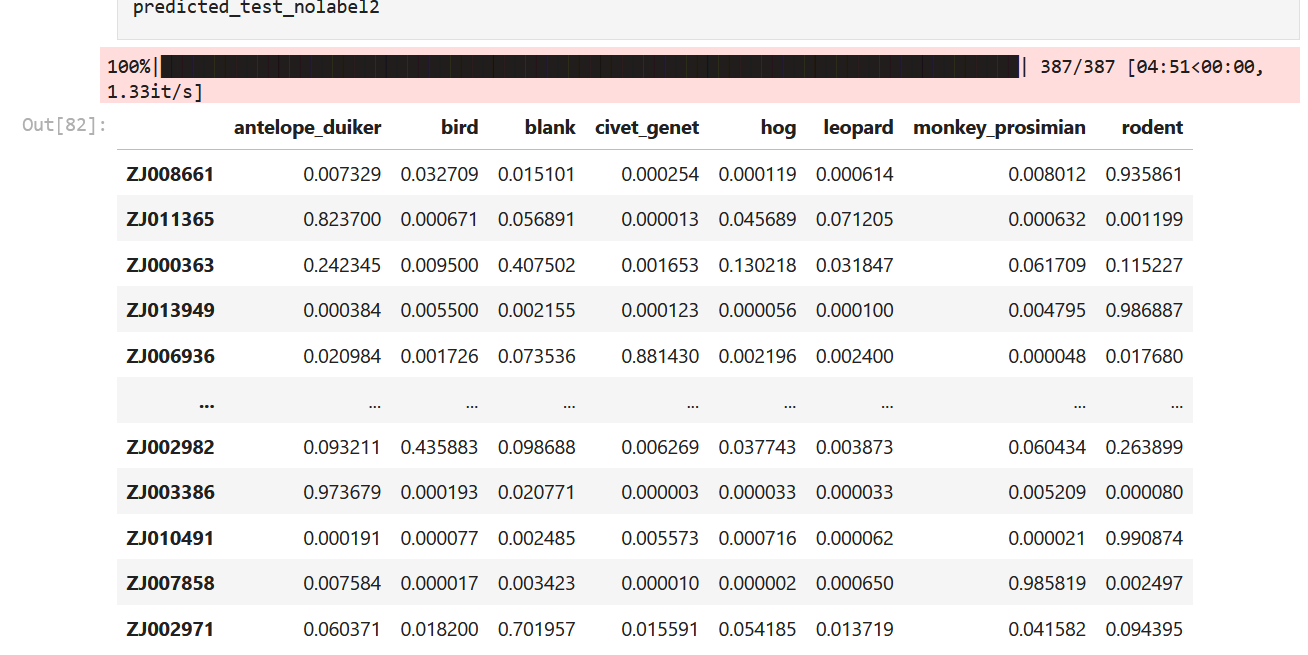


*Figure 12: Output vector of Model 1*



*Figure 12: Output vector of Model 2 (previous versions)*

Finally, we run the model with the test dataset, and also here we get very similar results, which seem not correct for Model 1. On the predicted dataset of Model 2 the code runs (Figure 13). On the later, as we have no label, we can pull out two images, one with a high score indicating clearly which species (Figure 14A) and one with a lower scores where the model is not clearly indicating a species (Figure 14B) from the code snippet below, in order to check if the predicted label corresponds to the species we would detect with human eye.



*Figure 13: Output vector of Model 2 – Predictions on Test Dataset*

|  |
| --- |
| *A.Highest Score indicated correctly Antelope* |
| *B.Model predicts blank with lower score* |

*Figure 14: Comparison of output with images*

# 6 Results Discussion

In our study, the ResNet50 model obtains a slightly performance and outperforms the AlexNet architecture. Model 1 needed however much more computational power to run through and during the study xxxx. However, due to the results as shown before, we failed to do a proper calculation of the relevant evaluation metrics, like Accuracy, Precision, Recall and F1, since it was not was not feasible.

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Precision | recall | f1 |
| Model 1: ResNet50 | N/A | N/A | N/A |
| Model 2: Alexnet | N/A | N/A | N/A |

*Table 1: Results comparison for Model 1 and Model 2 not feasible*

However, we would compute the how accurate the prediction is per category (Figure 2). We see the most common species like monkey and antelope achieve an accuracy of approximately 50%. More rare cases like bird and hog have comparably a poor accuracy.

|  |  |  |
| --- | --- | --- |
| Species | ResNet50 | AlexNet |
| Monkey proSimian | 50% | 49% |
| Antelope duiker | 50% | 49% |
| civet genet | 49% | 48% |
| leopard | 45% | 43% |
| rodent | 40% | 37% |
| bird | 33% | 33% |
| hog | 19% | 21% |
| blank | 45% | 45% |

*Table 2: Accuracy per category for both models*

Further it can be observed, that both models exhibited similar characteristics and challenges in terms of image recognition. Both have better performance for large animals such as Antelope Duiker and Leopard. For the smaller animals such as Rodent and Bird, the accuracy is lower. Also, we observe misclassification when images are blurred, which might be caused due to a fast movement of some animals, especially Birds (Figure 15A). Additionally, the models on some occasions misidentified the background as an animal (Figure 15B). A triggered camera trap seems to misclassify closely resembled animal forms from the static background of the image. Additionally, when animals were too close or too far, the models seem to have failed to clearly recognize the animal and usually indicated a score below 0.5 (Figure 15C and 15D). Overall, the results from the image recognition results of both models seem to be heavily affected when the quality of the image was rather poor (as also previously seen with Figure 14B).

|  |  |
| --- | --- |
| *A.Blurred image* | *B.Background as animal* |
| *C.Close-up of animal* | *A.Animal too far away* |

*Figure 15: Examples of Misclassifications*

Considering the low accuracy of both models, we might need to revisit the assumption, that there might be no effect of the dataset contained color images taken during the day and gray images taken during the night. Figure 16 illustrates that animals detected in night images are more confusing to the models, which is clearly seen in the output of the model, as the vector of probabilities is not clearly classifying the image to one of the species. Therefore, in future use for the model, it might be interesting to investigate further if the models would perform better when images from day and night were to be trained separately.

|  |
| --- |
| *A.Bird in night image misclassified as antelope* |
| *B.Bird in night vision misclassified as rodent* |

*Figure 16: Night images misclassified*

# 7 Conclusion and Outlook

During this project work, we trained and tested two CNN models on a camera trap image recognition. The models delployed show promise, however in terms of accuracy both models did not perform as well as we would have hoped.

On reflection, it seems we tried to deploy a quite complex model, which multi-class image recognition. In retrospective, we believe it would have been better to deploy a CNN model in the first place with a binary approach: filtering images which are blank. Then as a second step the model could predict which image contains which species.

Nevertheless, we believe that with improvements on the parameters and on the training dataset, we could improve the accuracy sufficiently. For instance, we might utilize more the preprocessing of the data to improve the image quality.

We could also customize the parameters of the models by fixing trainable and non-trainable parameters, apply more layers to the model.fc or applying filters might improve the accuracy. During this study, we opted for the Cross-Entropy Loss function, experimentation with other loss functions and different learning rates might improve the performance of the models as well.

Concerning the image recognition challenge, the most common models/architectures include the used ResNet and AlexNet, however there are also VGG-16 and the GoogleNet-2014,which are commonly used. VGG-16 is lower in complexity since it contains a relatively small number of sequential convolutional layers. ResNet, with its two versions ResNet50 and ResNet152, contains a deeper architecture than VGG-16, as it combines convolutional layers with residual modules. We only trained on ResNet50, therefore it would be interesting to train on ResNet152. Table 3 illustrates the most important networks for image recognition, which are worth exploring.

|  |  |  |
| --- | --- | --- |
| Model | Trained Layers | Description |
| VGG-16 | 16 | 13 convolutional layers, with a set of 3x3 filters, ReLu activation, followed by 3 fully connected layers |
| ResNet50 | 50 | 50 convolutional layers and 16 residual units |
| ResNet152 | 101 | 33 start modules and 99 convolutional layers, ReLu-based normalization and activation |
| AlexNet | 13 | 5 convolution layers, 3 max-pooling layers, 2 Normalized layers, 2 fully connected layers and 1 SoftMax layer, ReLu activation |

*Table 3: Most important Deep Learning Models for image recognition*

Perhaps most importantly, our results show that using machine learning and especially deep learning, a tremendous amount of time can be saved for researchers which do the labeling of the camera trap images manually. If further trained and accuracy improved, we believe that for wildlife detection and animal recognition, technology can save more than 90% of the human labor, assuming the 96.6% accuracy level of humas can be matched. Letting the technology do this time-consuming task of animal recognition, and redirecting the human labor to other important scientific purposes will accelerate research in this area.

Automatic image recognition may also benefit science projects, that are not able to recruit many human volunteers for supporting in this task; and drive cost reduction when it comes to gathering information from National Parks and Wild Habitats, hence enable more studies of animal and wildlife conservation.

# References

[1] Witmer, G. W. Wildlife population monitoring: some practical considerations. Wild. Res. 32, 259–263 (2005). [Google Scholar](https://scholar.google.com/scholar_lookup?title=Wildlife%20population%20monitoring%3A%20some%20practical%20considerations&journal=Wild.%20Res.&volume=32&pages=259-263&publication_year=2005&author=Witmer%2CGW)

[2] McEvoy, J. F., Hall, G. P. & McDonald, P. G. Evaluation of unmanned aerial vehicle shape, flight path and camera type for waterfowl surveys: disturbance effects and species recognition. PeerJ 4, e1831 (2016). [Google Scholar](https://scholar.google.com/scholar_lookup?title=Evaluation%20of%20unmanned%20aerial%20vehicle%20shape%2C%20flight%20path%20and%20camera%20type%20for%20waterfowl%20surveys%3A%20disturbance%20effects%20and%20species%20recognition&journal=PeerJ&volume=4&publication_year=2016&author=McEvoy%2CJF&author=Hall%2CGP&author=McDonald%2CPG)

[3] Köndgen, S. et al. Pandemic human viruses cause decline of endangered great apes. Curr. Biol. 18, 260–264 (2008). [Google Scholar](https://scholar.google.com/scholar_lookup?title=Pandemic%20human%20viruses%20cause%20decline%20of%20endangered%20great%20apes&journal=Curr.%20Biol.&volume=18&pages=260-264&publication_year=2008&author=K%C3%B6ndgen%2CS)

[4] Steenweg, R. et al. Scaling-up camera traps: monitoring the planet’s biodiversity with networks of remote sensors. Front. Ecol. Environ. 15, 26–34 (2017). [Google Scholar](https://scholar.google.com/scholar_lookup?title=Scaling-up%20camera%20traps%3A%20monitoring%20the%20planet%E2%80%99s%20biodiversity%20with%20networks%20of%20remote%20sensors&journal=Front.%20Ecol.%20Environ.&volume=15&pages=26-34&publication_year=2017&author=Steenweg%2CR)

[5] Kwok, R. Ai empowers conservation biology. Nature 567, 133–135 (2019). [Google Scholar](Kwok,%20R.%20Ai%20empowers%20conservation%20biology.%20Nature%20567,%20133–135%20(2019).)

[6] Burton, A.C., Neilson, E., Moreira, D., Ladle, A., Steenweg, R., Fisher, J.T., Bayne, E. & Boutin, S. Review: Wildlife camera trapping: a review and recommendations for linking surveys to ecological processes. Journal of Applied Ecology, 52, 675–685. . (2015)

[7] Beery, S., Van Horn, G. & Perona, P. Recognition in terra incognita. Proceedings of the European conference on computer vision (ECCV), pp. 456–473. (2018)

[8] Chen, P., Swarup, P., Matkowski, W.M., Kong, A.W.K., Han, S., Zhang, Z. & Rong, H. A study on giant panda recognition based on images of a large proportion of captive pandas. Ecology and evolution, 10, 3561–3573. (2020)

[9] Gomez Villa, A., Salazar, A. & Vargas, F. Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. Ecological informatics, 41, 24–32. (2017)

[10] A Swanson, et al., Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna. Sci Data 2, 150026 (2015). [Google Scholar](https://scholar.google.com/scholar_lookup?title=Snapshot+Serengeti%2C+high-frequency+annotated+camera+trap+images+of+40+mammalian+species+in+an+African+savanna&author=A+Swanson&publication_year=2015&journal=Sci+Data&pages=150026&doi=10.1038%2Fsdata.2015.26&pmid=26097743)

[11] Welbourne, D. J., A. W. Claridge, D. J. Paull, and A. Lambert. How do passive infrared triggered camera traps operate and why does it matter? Breaking down common misconceptions. Remote Sens. Ecol. Conserv. 2, 77–83. (2016) <https://doi.org/10.1002/rse2.20>

[12] He, K., Zhuang, X. Ren, S., Sun, J., Deep Residual Learning for Image Recognition. (2015) <https://arxiv.org/pdf/1512.03385.pdf>

[13] PyTorch Documentation, online. <https://pytorch.org/vision/stable/models.html>

[14] Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. How transferable are features in deep neural networks? In Proceedings of Advances in Neural Information Processing Systems (pp. 3320–3328). (2014) [Google Scholar](https://scholar.google.com/scholar_lookup?hl=en&publication_year=2014&pages=3320-3328&author=J.+Yosinski&author=J.+Clune&author=Y.+Bengio&author=H.+Lipson&title=Proceedings+of+Advances+in+Neural+Information+Processing+Systems)