NOSpcimen: A first approach to unsupervised discarding of empty photo trap images

David de la Rosa[1], Antón Álvarez[2], Ramón Pérez[2], Germán Garrote[3], Antonio J Rivera[1[0000−0002−1062−3127]], María J del Jesus[1[0000−0002−7891−3059]], and Francisco Charte[1[0000−0002−3083−8942]]

[1] Dept. of Computer Science, University of Jaén. Spain
(drrosa,arivera,mjjesus,fcharte)@ujaen.es
[2] WWF Spain, Spain
(aalvarez, rapayal)@wwf.es
[3] Consejería de Medio Ambiente y Agua. Junta de Andalucía
German.garrote.alonso@juntadeandalucia.es

Abstract. A key tool in wildlife conservation is the observation and monitoring of wildlife using photo-trapping cameras. Every year, thousands of cameras around the world take millions of images. A large proportion of these are empty – they do not show any animals. Sorting out these blank images requires considerable effort from biologists, who spend hours on the task. It is therefore of particular interest to automate this task. So far, systems have been proposed which are based on the use of supervised learning models. In order to learn, these systems require the annotation of images to indicate where animals are located within them. NOSpcimen (NOn-SuPervised disCardIng of eMpty images based on autoENcoders) system takes a different approach. It relies on unsupervised learning mechanisms. Thus, no prior annotation work is required to automate the process of discarding empty images.

Keywords: Unsupervised learning, deep learning, neural networks, clustering, robust autoencoder

1 Introduction

Monitoring the natural environment is vital to the preservation of species, but it is not a trivial task. It requires professionals capable to recognize the presence of animals and time to understand their behavior [1].

The use of photo-trapping cameras [2] is usually one of the best options to solve this problem. A photo-trapping camera is a tool for the surveillance and study of the wild environment from a fixed installation. These devices provide data on the location, population, size and interaction of species. The camera has a motion sensor connected to it, so that when an animal moves close to the sensor it causes the camera to fire, recording a burst of images without interfering with the animal's behavior.

One of the major disadvantages of this technique is the problem of blank images. A large proportion of the captured images are empty, they do not show any animals. Many situations can cause this problem, such as the motion sensor detecting the movements of an inanimate object, the animal passing by too quickly to be captured by the camera or the negative impact of high temperatures on motion sensors. For this reason, the use of software that automatically filters out empty images is a useful tool for many professionals in this field.

Machine learning (ML), a subfield of artificial intelligence, focuses on the creation of systems able to learn and adapt to the problem on the basis of the training data provided. Regarding to this problem, the use of ML implies the creation of a training dataset of images to develop models for the detection of empty and non-empty images.

Most of the ML proposed systems follow a supervised approach, that require the annotation of images in the training phase (which is expensive to manufacture since it needs human actions to identify, categorize and annotate the data) to indicate where animals are located within them. Then, the model compares the prediction and the actual value and use that information to infer knowledge that will allow them to recognize patterns in non-labelled images.

NOSpcimen takes an unsupervised approach that avoids the required image annotation work, apart from a previous separation of empty and non-empty images. The idea is to group the set of images according to their features, train an autoencoder (AE) for each cluster to reconstruct empty images and train a Multilayer Perceptron neural network (MLP) to classify images as empty or non-empty by taking the reconstruction errors.

The rest of this paper is structured as follows. Starting with preliminaries in section 2, NOSpcimen's details are explained in section 3. Section 4 focuses on the experimentation carried out in order to optimize the models and on the results achieved. Lastly, section 5 explains the conclusions reached after the completion of the project.

2 Preliminaries

Many solutions can be found in the literature to solve the empty images filtering problem. They are grouped into two main areas.

On the one hand, solutions based on computer vision. The study [3] assumes that when the camera trap detects activity, it takes photographs with the same background. The only difference would be found in the movement of the animal. If it were a set of blank images, there would be no difference between them.

On the other hand, deep learning-based solutions, where a neural network is trained to differentiate between empty and non-empty images. For example, the study [4] trained a convolutional neural network using the ResNet-18 architecture and more than three million images for identification of wildlife in camera trap images, achieving an accuracy between 82 and 98%, depending on the test set. Also, [5] follows a similar scheme, training very deep convolutional neural network for automatic species classification.

Another supervised example is MegaDetector [6], a free tool to classify images as it contains animals, vehicles, persons or if it is empty. The latest version, the fifth, incorporates the YOLOv5 network, allowing users to classify their own images with high processing speed (up to three times faster than the previous version that used the Faster-RCNN architecture). This model is trained on bounding boxes from a variety of ecosystems with both private and public data.

As previously described, these studies follow supervised approaches. To the best of our knowledge, there are not unsupervised approaches for the empty images filtering problem on photo trap images.

Apart of deep learning models such as those described, it is important to define the concept of AE. An AE [7] (see figure 1) is a type of neural network that aims to reconstruct the input of the network onto the output, creating constraints so that the output is not a direct copy of the input. This is achieved by creating a coding layer, which represents the encoding of the input given by the encoder layers. Then, the decoder layers reconstruct the input based on its latent representation in the encoding.

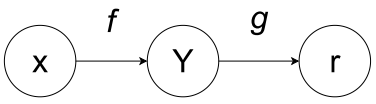

Fig. 1: Structure of an AE. x represents the input, f the encoder, Y the encoding, g the decoder and r the output

Several types of AE can be found in order to improve the results of a basic structure for different problems [7]. For example, denoising or robust autoencoders (RAE) are more tolerant to noisy inputs, sparse autoencoders include a penalty in the learning process and convolutional autoencoders use convolutional layers, which works properly with images as input.

Focusing on RAE models [8], the main difference with a classic AE resides in the loss function used. Correntropy based loss function [9] allows RAE to be more tolerant to noisy inputs. The loss function is presented in equations 1 and 2, where y_{pred} represents the RAE prediction, y_{true} the actual value and σ a constant that represents the kernel size.

$$Loss = -\sum kernel(y_{pred} - y_{true}) \tag{1}$$

$$kernel(\alpha) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{\alpha^2}{2\sigma^2}) \tag{2}$$

The presented AE architectures are used in many different problems, such as data compression [10], dimensionality reduction and data visualization [11], data denoising [12] or even as a module to solve an anomaly detection problem, as will be shown in section 3.

3 Our proposal: NOSpcimen system

The empty images filtering problem fits well as a classification problem. Nevertheless, NOSpcimen aims to transform the classification task into an anomaly detection problem where images containing some animals are considered as anomalies. The survey in [13] presents an overview of research methods in deep learning-based anomaly detection. Focusing on unsupervised deep anomaly detection, AE are the most common used architecture.

In this line, our working hypothesis is as follows. An AE is able to reconstruct the input onto the output. Therefore, if an AE is trained to only reconstruct empty images, the reconstruction error will be higher when passing a non-empty image as input to the model. Evaluating the reconstruction error, e.g. using a neural network, we may be able to distinguish if an image contains animals.

Nonetheless, the set of images are highly variable depending on the time of the day when the photography was taken (daytime or night), the brightness of the environment or the characteristics of the terrain (with heavy vegetation or in dry environments). This results in an AE having to learn many different image patterns, which causes lower accuracy in the reconstruction process. For this reason, we propose separating the dataset into several groups of images with similar features, so that one AE would be trained for each group, specializing in the characteristics of that group.

By joining all these blocks together, NOSpcimen is proposed as an unsupervised solution for the empty images filtering problem. Its diagram is presented in figure 2. Details are provided in the following subsections.

3.1 Image segmentation using K-means clustering

A clustering algorithm was trained by using empty images, creating N groups of images based on the image histogram, a representation of the number of pixels in an image as a function of their intensity.

Specifically, we have implemented Lloyd K-means algorithm [14]. Each image is assigned to a cluster by measuring the distance to the center of each one, the centroids, and choosing the closest group.

3.2 Robust autoencoders for image reconstruction

As described in section 2, RAEs are more tolerant to noisy inputs so that it has been decided to use RAEs instead of vanilla AEs.

Once the image clusters have been determined, the next step is to create the RAE models, one for each group of images trained only with his own empty images in order to optimize their reconstruction.

Following this logic, the reconstruction of images where animals appear should be worse, so measuring the reconstruction error we can determine the presence of animals in each image.

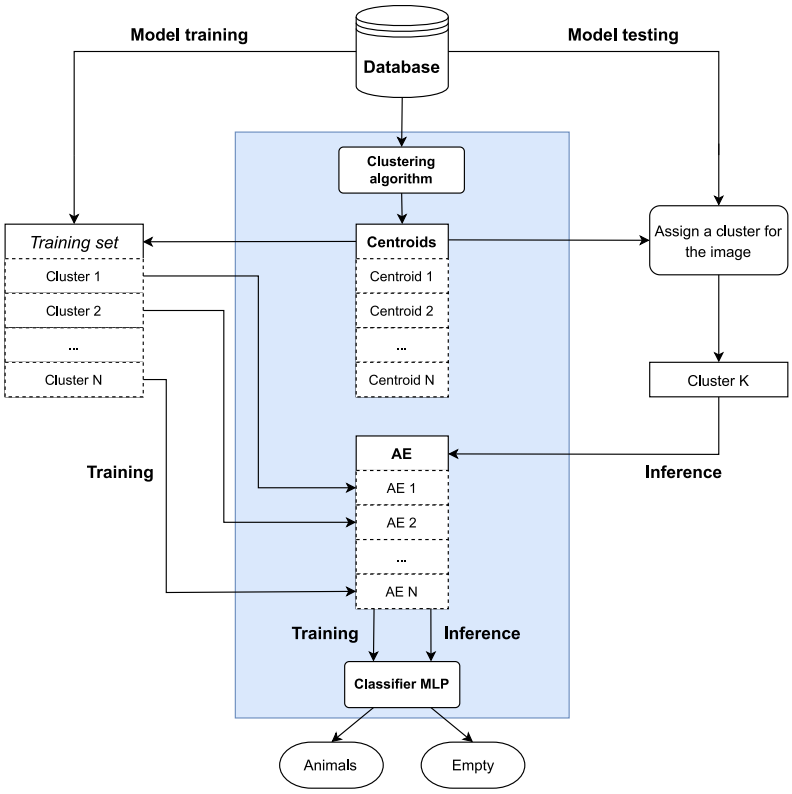

Fig. 2: Diagram of the proposed system

3.3 Multilayer-Perceptron Neural Network for image classification based on reconstruction error

The last step is to develop a method to classify images based on the difference between original and reconstructed images given by RAEs. To measure that difference, three metrics have been used: mean squared error (MSE) and mean absolute error (MAE), where m is the image width, n the image height, Y the original image and $\hat{Y}$ the reconstructed image, and structural similarity (SSIM), where x is the original image, y the reconstructed image μ is the average of the image, σ^2 is the variance of the image, σ the covariance of the image, $c_1 = (k_1 L)^2$ and $c_2 = (k_2 L)^2$, where L is the dynamic range of the pixel values, $k_1 = 0.01$ and $k_2 = 0.03$ by default (see equations 3, 4 and 5).

$$MSE = \frac{1}{m * n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [Y(i,j) - \hat{Y}(i,j)]^2 \tag{3}$$

$$MAE = \frac{1}{m * n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} |Y(i,j) - \hat{Y}(i,j)| \tag{4}$$

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{5}$$

In this line, we decided to create a neural network model capable to do that. Concretely, using MSE, MAE and SSIM values and the cluster identifier associated to the image as inputs, an MLP was trained to determine the presence of animals in each image based on the reconstruction errors. This network was trained with reconstruction error of both empty and non-empty images.

As summary, the creation of the model follows the steps below:

1. Creation of N groups of images with similar features.
2. Development of a set of RAEs specialized in the characteristics of each cluster trained to reconstruct blank images.
3. Development of a MLP network to classify the images as empty or non-empty taking as input the difference between original and reconstructed images.

The classification of new images consists of the following steps:

(a) Determine the appropriate cluster for the input image using centroids obtained in step 1.
(b) Reconstruct the image using the RAE associated with the cluster.
(c) Calculate the reconstruction error between the reconstructed and original image and predict the presence of animals using the MLP network.

4 Experimentation

In this section how NOSpcimen has been empirically tested is explained. Section 4.1 introduces the dataset and the preprocessing of the images. After that, the training process of every model will be analyzed, focusing on clustering in section 4.2, on RAE models in section 4.3 and ending up with MLP classifier in section 4.4. Finally, obtained results are presented in section 4.5.

4.1 Dataset

The photo trap images we will work with belong to WWF Spain. We have a total of 92 573 images, of which 37 503 are empty and 55 070 contains some animals.

To prepare data for model input, preprocessing is necessary. Original images are 2560 pixels wide and 1920 pixels high, with a depth of three bits per pixels, since they are RGB images. Those dimensions are excessive for deep learning models for several reasons, such as limited RAM memory or execution time. Moreover, large dimensions do not imply better results. Hence size reduction is necessary. After a visual analysis of a random subset of images applying a

scaling in the image width to 512, 384, 256 and 192 pixels, we decided to scale the images to 384 pixels wide and 288 pixels high, maintaining the same depth.

In addition, all the camera trap images have timestamps at the bottom that include the date and time the photograph was taken. Removing this data reduces the input noise of the neural network.

Finally, the train test validation split is performed by randomly choosing images for empty and non-empty classes following the proportion 60-20-20, 60% images for training, 20% for validation and 20% for testing.

4.2 Clustering algorithm

Two different implementations of the K-means algorithm were chosen: Scikit-learn version [15] and NLTK version [16]. The main difference resides in the distance metric: while the former only allows euclidean distance, the latter provides many other metrics, as cosine, Manhattan or correlation distances.

As stated in section 3.1, clustering algorithm will be based on the use of the histogram of the images, which provides a good representation of the inner properties of the images. The number of bins for the histograms will be (*85x85x85*), grouping the pixels values into bins of size three.

Different parameters were analyzed to get the best combination: color model (RGB and HSV), distance metric (euclidean, cosine, Manhattan and correlation distance), and number of clusters that will be formed (between 1 and 12). Results were measured using intra-cluster distance (ICD) obtained.

Color model	**ICD**
RGB	770.05
HSV	1 157.47

Table 1: Color model experiments.

Distance metric	**ICD**
Euclidean	770.05
Canberra	820.86
Cosine	856.57
Manhattan	902.32

Table 2: Distance metric experiments.

As shown in tables 1 and 2, RGB color model contributes less ICD than HSV. Euclidean distance is also the best distance metric in terms of ICD.

Concerning the number of clusters, figure 3 shows a steeper slope between six and seven clusters, from which the slope is much smoother.

Because of presented results, we decided to use RGB color model, euclidean distance and seven clusters of images.

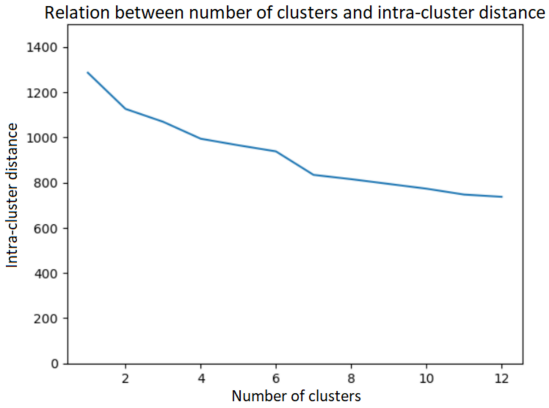

Fig. 3: Relation between number of clusters and intra-cluster distance.

4.3 Robust autoencoders

RAE models decisions are focused in two aspects: model architecture and model parameters.

As to model architecture, after some testing, the chosen one is shown in figure 4. Two tests were created in order to verify that it gives optimal results. On the one hand, reducing the size of the network worsens the results as it does not have the capacity to process the information provided by the images. On the other hand, increasing the size of the model by adding more layers and reducing the encoding layer's size also worsen the results. If the encoding layer is excessively small, the decoder network will not be able to reconstruct the image properly.

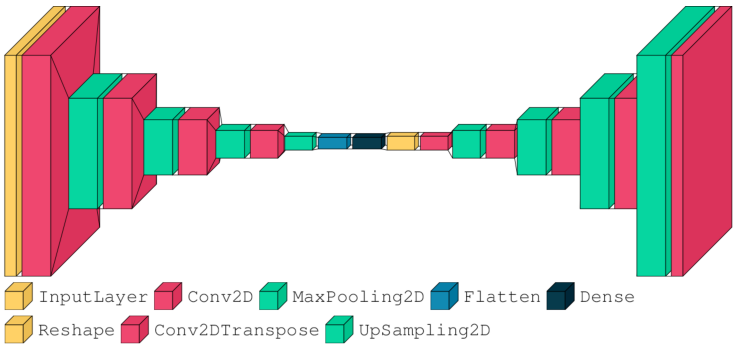

Fig. 4: Architecture of RAE chosen.

Table 3 resumes the model parameters, settled based on loss and MSE functions obtained during the model training (see figure 5). The figures show a stag-

nation from 70 epochs onwards in both loss and MSE functions, which proves that choosing more than 70 epochs is a bad option. Some testing was made in order to determine if a lower number of epochs is better. Nevertheless, these tests gave slightly worse results, so we decided to stick with the initial option. Optimizer and learning rate parameters have default values.

All RAEs maintain the same architecture and parameters.

Parameter	Value
Epochs	70
Batch size	16
Optimizer	Adam
Learning rate	0.001

Table 3: RAE chose parameters

Parameter	Value
Epochs	120
Batch size	16
Optimizer	Adam
Learning rate	0.001
Input	MSE, MAE, SSIM, CID

Table 4: MLP chosen parameters.

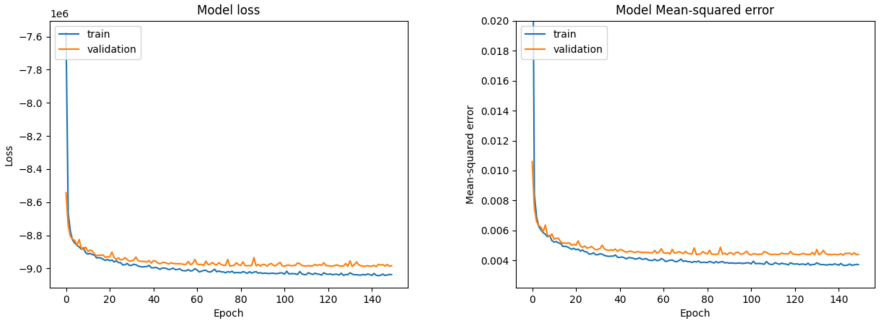

Fig. 5: Loss and MSE functions evolution during model training.

4.4 Multilayer-Perceptron classifier

Following the same scheme of RAE models, the MLP experimentation will focus in model architecture and model parameters.

The input of the MLP network is simpler than RAEs, as it will only be composed of three different error metrics between reconstructed and original image and one integer representing the cluster identifier. Therefore, the model architecture will be also simpler. It is formed by two densely connected layers of 20 neurons and a softmax output layer that allows to classify the input as an animal or empty observation.

Model parameters are presented in table 4. The network was trained based on cross-entropy loss between predicted and true labels, which calculates a score that summarizes the average difference between the actual and predicted probability distribution for each input. A test was performed in order to determine whether including the cluster identifier (CID) in the input layer improved the results. As shown in figure 6, accuracy is slightly improved when including CID. Also, accuracy value stagnates in $0.88 - 0.89$ from 120 epochs onwards.

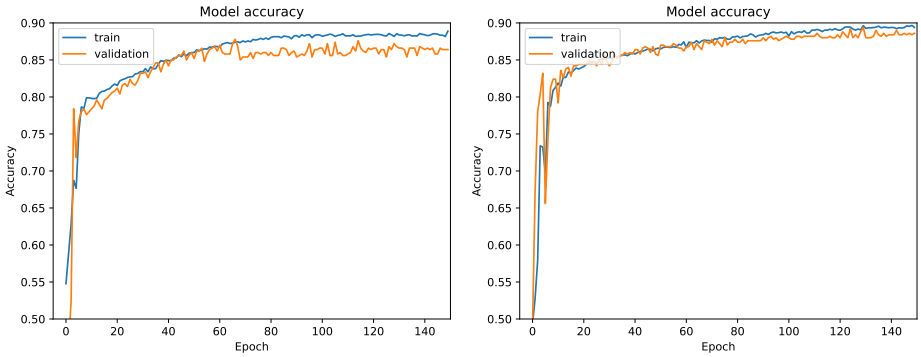

Fig. 6: MLP accuracy comparison including cluster identifier (right) or not (left)

4.5 Results

This subsection covers the results achieved by NOSpcimen on the test set. Figure 7 presents the confusion matrix and the ROC curve. Moreover, table 5 summarises the classification performance. The confusion matrix shows a majority of correct predictions (top left corner and bottom right corner) compared with incorrect ones. This is also endorsed by the 0.89 F1-score. AUC is also a good metric to prove the proper performance of the system, with a value of 0.955, near its maximum possible value.

Overall, given its predictive performance, we can state that the developed software fulfill the proposed objectives.

Finally, a comparison between Megadetector v5a and NOSpecimen using the same image test set is also presented in table 5. Megadetector confidence threshold was set at 0.2.

As can be seen, Megadetector's results are slightly better than those of NOSpcimen in terms of F1-score and AUC. Nevertheless, it is remarkable that Megadetector is a supervised approach, trained with much more data than NOSpcimen (various millions for the former against just 40 000 images for the latter). This fact greatly simplify the training phase, as in a supervised approach it is necessary to dedicate a time to annotate the data while this is avoided in a non-supervised approach.

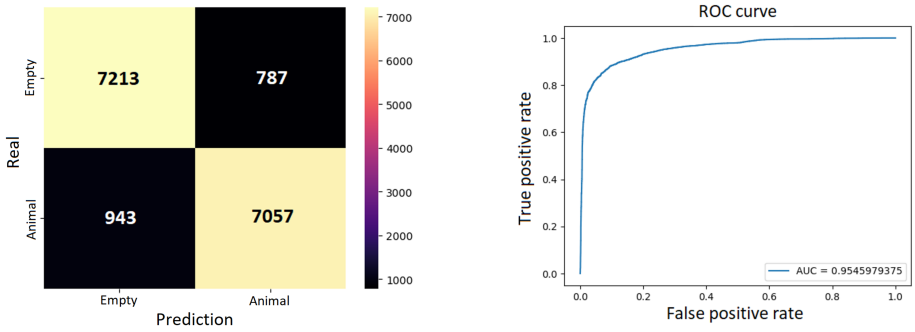

Fig. 7: Confusion matrix and ROC curve achieved.

Class	**Precision**	**Recall**	**F1-score**	**AUC**	**Accuracy**
Empty	0.884	0.902	0.892	-	-
Animals	0.900	0.882	0.890	-	-
Global	0.892	0.892	0.891	0.955	0.892
Megadetector	0.955	0.955	0.960	0.991	0.960

Table 5: Precision, recall, F1 score, area under ROC curve and accuracy values

5 Conclusions

With the results presented in the previous section, we can state that the proposed objectives have been achieved. The effort to empty images filtering task could be notably reduced by using NOSpcimen, allowing the professionals to focus only on non-trivial images, those that contain animals, instead of dedicate valuable time to the manual filtering.

In that line, a web application has been developed in order to make an usable prototype of this software. Figure 8 shows the main screen of the application (since the application will be used by Spanish conservationists, the user interface is in Spanish). Some parameters can be modified in order to adjust the software to the users' needs, as if they want to copy or move the images, if they want to store dubious images, etc. In addition, a threshold value can be seen in the image, which can take values between [0.5, 1]. The model set a probability for each image to be empty or not, in a range [0,1]. In case that the user do not need to store dubious images, each image will be assigned to the more probable class. On the other hand, if dubious images are stored, only images that exceed the threshold value will be assigned to a class. The rest will be assigned to *dubious* class. Following this reasoning, a high threshold implies that many images will be classified as dubious but with a low false positive or false negative rate, while a low threshold implies fewer images will be classified as dubious but with more false positives or false negatives.

The conservationists of WWF Spain are currently testing and validating this prototype with their own datasets. They plan to integrate the stable product as a component in their workflow being developed to optimize the monitoring of the Iberian lynx using camera trapping.

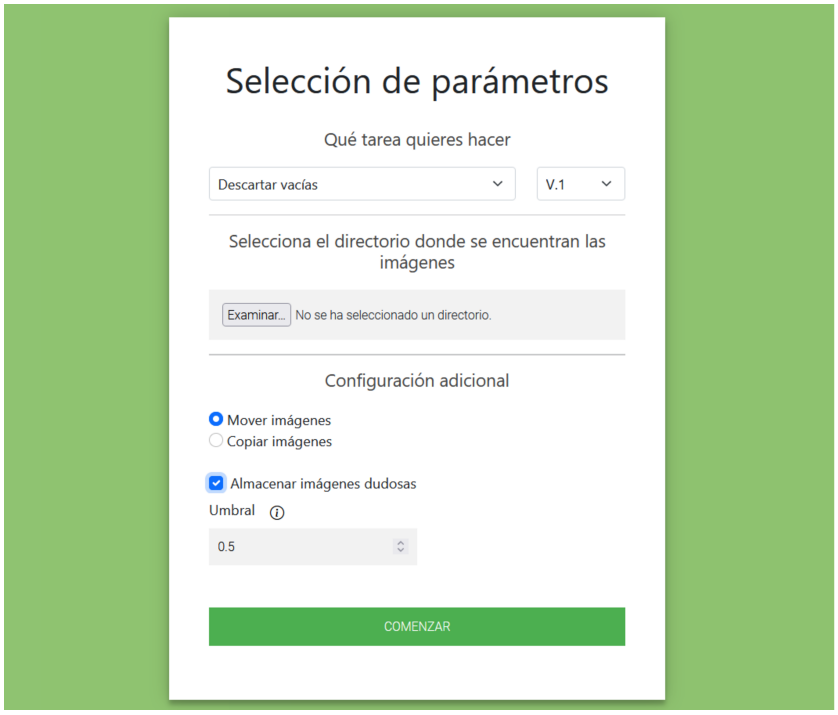

Fig. 8: Main page of the application.

Lastly, as seen in subsection 4.5, considering that this is a first approach to unsupervised discarding of empty photo-trapping images, our results are close to those of other supervised and state-of-the-art approaches, proving that unsupervised learning have the potential to provide valuable insights into biodiversity conservation problems. We are currently working to introduce some changes to improve this prototype. One of our objectives is to use other types of autoencoders, such as masked autoencoders [17], to increase the distance between reconstruction errors of empty and non-empty images. A clustering algorithm improvement is also planned to get more homogeneous groups of similar size by trying other distance metrics.

Acknowledgements

The research carried out in this study is part of the project "ToSmartEADs: Towards intelligent, explainable and precise extraction of knowledge in complex problems of Data Science" financed by the Ministry of Science, Innovation and Universities with code PID2019-107793GB-I00 / AEI / 10.13039 / 501100011033. Also, this work was partly enabled by Antón Alvarez's participation in the CV4Ecology Summer Workshop, supported by the Caltech Resnick Sustainability Institute.

References

1. Devis Tuia, Benjamin Kellenberger, Sara Beery, Blair R Costelloe, Silvia Zuffi, Benjamin Risse, Alexander Mathis, Mackenzie W Mathis, Frank van Langevelde, Tilo Burghardt, et al. Perspectives in machine learning for wildlife conservation. Nature communications, 13(1):792, 2022.
2. Natasha De Bondi, John G White, Mike Stevens, and Raylene Cooke. A comparison of the effectiveness of camera trapping and live trapping for sampling terrestrial small-mammal communities. Wildlife research, 37(6):456–465, 2010.
3. Weideng Wei, Gai Luo, Jianghong Ran, and Jing Li. Zilong: A tool to identify empty images in camera-trap data. Ecological Informatics, 55:101021, 2020.
4. Michael A Tabak, Mohammad S Norouzzadeh, David W Wolfson, Steven J Sweeney, Kurt C VerCauteren, Nathan P Snow, Joseph M Halseth, Paul A Di Salvo, Jesse S Lewis, Michael 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.
5. Alexander Gomez Villa, Augusto Salazar, and Francisco Vargas. 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.
6. Sara Beery, Dan Morris, Siyu Yang, Marcel Simon, Arash Norouzzadeh, and Neel Joshi. Efficient pipeline for automating species id in new camera trap projects. Biodiversity Information Science and Standards, 3:e37222, 2019.
7. David Charte, Francisco Charte, Salvador García, María J del Jesus, and Francisco Herrera. A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines. Information Fusion, 44:78–96, 2018.
8. Yu Qi, Yueming Wang, Xiaoxiang Zheng, and Zhaohui Wu. Robust feature learning by stacked autoencoder with maximum correntropy criterion. In 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP), pages 6716–6720. IEEE, 2014.
9. Weifeng Liu, Puskal P Pokharel, and Jose C Principe. Correntropy: A localized similarity measure. In The 2006 IEEE international joint conference on neural network proceedings, pages 4919–4924. IEEE, 2006.
10. Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy image compression with compressive autoencoders. arXiv preprint arXiv:1703.00395, 2017.
11. Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006.

12. Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. Advances in neural information processing systems, 25:341–349, 2012.
13. Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407, 2019.
14. Stuart Lloyd. Least squares quantization in pcm. IEEE transactions on information theory, 28(2):129–137, 1982.
15. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. the Journal of machine Learning research, 12:2825–2830, 2011.
16. Steven Bird, Ewan Klein, and Edward Loper. Natural language processing with Python: analyzing text with the natural language toolkit. O'Reilly Media, Inc., 2009.
17. Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 16000–16009, June 2022.