Detection and Counting of Animals in Camera-Trap Images using Faster R-Cnn and Data Augmentation Techniques

Panawè Touh, Mukesh Sharma

Abstract: ***Camera traps are used to recover images of animals in their habitats to help in the conservation of fauna. Millions of images are captured by camera traps and extracting information from these data delays and consumes enough resources so sometimes millions of images cannot be used due to lack of resources. That is why researchers have proposed solution approaches using Convolutional Neural Networks (CNNs) and object detection models to be able to automate the retrieval of information from these images. We used Faster R-CNN and data augmentation techniques on Gold Standard Snapshot Serengeti Dataset to detect animals in images and count them. The performances of the two models (the one trained on the original dataset and the one trained on the augmented dataset) were compared to show the importance of having more data for this task. Using the augmented dataset, we trained our model which reached an accuracy of 98.26% for classification of the proposed regions, an accuracy of 79.55% for counting the species present on the images and a mAP of 95.3%. For future work, the model can be trained to recognize the actions and characteristics of animals and tuned to be more efficient for counting task.***

Keyword: ***Camera trap, Object detection, Faster R-CNN, Data augmentation***

I. INTRODUCTION

It is widely assumed that with the use of camera traps and especially with their proliferation in many projects which can be counted per hundred, we have better understood the natural environment and species with the data collected [1]. Even with almost perfect efficiency, camera traps pose a major problem for researchers because they generate a lot of images so significant resources such as time and manpower are used to analyse all of these images in order to extract information necessary for studies. As result, sometimes some photos are not used for research since information could not be extracted. With the development of very efficient techniques and models in the field of deep learning, automatic information extraction solutions have been proposed to overcome this problem. Deep learning models gave the ability to computers to analyse and understand digital images [2].Identifying and collecting information for each object in images is more beneficial and provides more information than simply classifying images into a class [3]. Image classification models have shown their limits in retrieving individual information from objects in images when object detection models shown to be more efficient. To automate these tasks, we opted for an object detection model, Faster R-CNN [4], which showed satisfactory performance and outperformed the other models in terms of accuracy and combined execution time. For these tasks, it will be necessary to exceed human accuracy for the extraction of information in order to provide a precise and efficient system. Previous research may have exceeded or reached the accuracy of classification of animals by humans, but the performance of the various systems proposed on other tasks such as recovering the number of animals present in the images, their characteristics (young or not) or their actions (eating, resting, moving ...) are less excellent than human performance. The goal of the present article is therefore to propose a method in order to have a sufficiently powerful model to classify the animals present on the images captured by the trap cameras and to count them.

For this purpose, we used the Faster R-CNN [4] object detection model that we trained on a reduced dataset from the Snapshot Serengeti dataset [5]. The Snapshot Serengeti project from season one to eleventh collected 7.1M images and thanks to the volunteers, the images were tagged for 61 animal species and 78,000 images were annotated with 150,000 bounding boxes [6]. Due to lack of resources (computing power, storage memory), we therefore rather used the Gold Standard Snapshot Serengeti Bounding Box Coordinates dataset [7] which is a dataset of 4010 images of Snapshot Serengeti dataset with the coordinates of their objects. We have noticed that the performance of the model trained on the Gold Standard Snapshot Serengeti Bounding Box Coordinates dataset is not efficient. We therefore used data augmentation techniques in order to make the dataset more complex and large for better results. The MMDetection object detection framework [8] was used in our task to train and test our models. Scripts have been written for tasks like manipulating our dataset and counting detected objects.

II. RELATED WORK

Several before us have tried to provide solutions to this problem. Two types of methods have been observed. Some solutions have used just Convolutional Neural Networks to extract and give the information contained in the images. Other solutions have used the techniques of object detection which consists in first detecting the objects in the images and then classifying them.
[9], [10], [11], [12],

Revised Manuscript Received on June 15, 2020.
* Correspondence Author
Panawè Touh*, CSE Department, AP GOYAL SHIMLA UNIVERSITY, Shimla, India. E-mail: magloiretouh@gmail.com
Mukesh Sharma, CSE Department, AP GOYAL SHIMLA UNIVERSITY, Shimla, India. E-mail: sharmamukesh8113@gmail.com

[13] used ConvNets to try to automate information retrieval.

Through the use of ConvNets (AlexNet [14], VGGNets [15], GooLeNet [16] and ResNets [17]), Alexander Gómez et al. in [9] wanted to show how the Deep CNNs will impact the classification of animal species in the images captured by camera traps. Given the imbalanced nature of the Serengeti dataset Snapshot, they created sub datasets (unbalanced, balanced, conditioned and segmented) to train the deep CNNs in order to see the importance of the data used to train the models. They noticed that with a well-balanced dataset and images cropped and necessarily containing animals or their parts, the ConvNets perform better.

Still in this perspective of seeing the importance of using a well-balanced dataset in the classification of animals on images, Lewis Guignard and Noam Weinberger in [10] have extracted 300 images of the two species which are zebras and gazelles in the Snapshot Serengeti dataset and 300 images without images to train a Multi-Layer Perceptron (MLP) Network. Their model achieved 92% accuracy for the classification task.

To detect if an image is empty and to automatically recognize the three most common species (bird, bandicoot, and rat) in their dataset, Nguyen et al. in [11] using deep ConvNets like AlexNet, VGG-16, GooLeNet, and ResNet-50 have reached an accuracy of 96.6% for the first task which is whether the image is empty or not and 90.4% for the second task which is the recongnition of species.

Much like the previous solutions, Tabak et al. in [12] trained a deep ConvNet (ResNet-18) on millions of images from different camera traps projects in the United States and then tested it on camera traps datasets from Canada, Tanzania and part of the dataset of USA. The model has shown a good classification performance by obtaining a top-1 accuracy of 98% on the dataset of the USA, 94% on that of Tanzania and 82% on that of Canada.

Mohammad S. Norouzzadeh et al. in [13] separated the problem into two distinct phases. The first phase is to say whether or not an image contains animals and the second phase is to say what kind of species are present, in what number and their descriptions. Using 9 CNNs (AlexNet [14], NiN [18], VGG [15], GoogLeNet [16] and ResNet 18, 34, 50, 101, 152 [17]) and the multitask learning technique [19], they were able to train both phases in a single model on the Snapshot Serengeti dataset containing 3.2 million images. ResNet-101 and ResNet-152 are the models with the best results whether in classification or counting and description. ResNet-101 had a top-1 accuracy of 93.8% and a top-5 accuracy of 98.8% for the classification task and a top-1 accuracy of 61.4% and a top-5 accuracy of 83.4% for the counting task. ResNet-152 performed exactly the same as ResNet-101 for the classification task and a top-1 accuracy of 62.8% and a top-5 accuracy of 83.6% for the counting task.

[20], [21], [22] have used an object detection approach. Hayder Yousif et al. in [20] first detect objects in the foreground using an effective dynamic background modelling and then classified them into three classes (human, animal, background patches) using AlexNet [14]. They reached an accuracy of 93.4% for the classification of species.

Schneider et al. in [21] compared the performances of two object detection models which are Faster R-CNN [4] and YOLOv2 [23] on images captured by camera traps. They noticed that the Faster R-CNN model has a better performance with an accuracy of 93% compared to YOLOv2 with an accuracy of 76.7% even if YOLOv2 is faster.

For their work, Ahmed et al. [22] have detected animal positions using K-mean [24] clustering and graph cut [25] and then classified these regions into species categories with a deep neural network. They reached 99.75% for the detection of the regions and 90.89% for the classification of these regions in categories of species.

According to previous work, even if ConvNets sometimes have good accuracy for classification, they do not take into account other information such as the age and characteristics of the animals that researchers need. It because even with a very large dataset, Mohammad S. Norouzzadeh et al. in [13] could not have had very good results as regards the tasks of counting and description of the animals. Object detection techniques are much more suitable for this kind of problem and the Faster R-CNN [4] model is one of the most efficient therefore more likely to have good performance. We also noticed that the data used to train the models play an important role. The more the datasets are well balanced or have many data for each class, better performances are.

III. OBJECT DETECTION MODELS

A. Deep Learning

With the data boom in our era, deep learning methods perform better than state-of-art machine learning methods in several areas, especially in computer vision with complex data such as images [26]. Deep learning models, based on layers, manage to extract complex information from a set of data, which then allows them to do mapping in order to classify this data into several classes. To do this, deep learning models imitate the functioning of the brain with layers of neurons linked together to pass information for processing. McCulloch and Pitts in 1943 tried to understand the functioning of biological neurons in order to develop an artificial neuron capable of receiving and processing information, in order to subsequently allow the construction of more complex patterns [27]. Their work has revolutionized the field of artificial intelligence. Based on their work, improved neurons and artificial neural networks have been developed. A Convolutional Neural Networks (CNNs) are neural networks that use convolutional layers to extract features and patterns from data, often images. The most popular are LeNet [28], AlexNet [14], VGG [15], GoogLeNet [16], ResNet [17]. They have proven their performance in recognizing patterns hidden in data. CNNs are extremely successful in computer vision applications, such as face recognition, object detection, powering vision in robotics, and self-driving cars.

B. Object Detection

Object detection is a problem of computer vision which encompasses localization and classification [29]. Both localization and classification techniques are used to identify the position and class of the various objects present on an image or video. Object detection models are used to retrieve much more information in order to better understand an image. For an object detection problem, object detection methods have almost the same process for locating and classifying objects. Region generation techniques are used to provide a set of regions, then feature extraction models are used to extract the features of the regions proposed in the first step and finally the regions are classified using classification techniques. To avoid the fact that several regions refer to the same object, techniques such as NMS and soft-NMS [30] are used to reduce and keep the most optimal detections for each object. Region-based models with CNNs have shown performance above other models in term of accuracy but there are several models that outperform them on processing speed because they do real time. Object detection models using CNN are separated into two types: one stage models and two stages models.

C. One Stage Models

One stage models are mostly used for real time needs. They are faster compared to two stage models because they detect and classify regions in a single pass. One stage models approaches adopt the idea of regression, that is dividing the input image into several cells and each cell predicts boxes and class probability. YOLO (You Only Look Once) is a detection model developed by Joseph Redmon et al. whose first version YOLOv1 [31] quickly won over thanks to its real-time processing (45 frames per second) but the problem with the first version of YOLO is that it can only detect one object per cell and has poor accuracy for detecting small objects and obstructed objects. YOLO has successfully posed the detection problem as a regression problem which, starting from the image and in a single pass, can detect objects and their classes. YOLOv2 [23] has improved the detection speed (up to 67 frames per second) and the accuracy which at 40 frames per second exceeds the Faster R-CNN [4] model for the PASCAL VOC 2007 dataset [32]. YOLOv3 [33] made it possible to detect three objects per cell which help to improve the accuracy. SSD [34] is a detection model based on the idea of YOLO regression and the idea of Faster-RCNN anchors to be able to make predictions with high accuracy in real time. The disadvantage of one stage models is their relatively low accuracy compared to two stage models.

D. Two Stage Models

To propose a new way of operating in order to change the performance of object detection models, Ross Girshick et al. have developed the R-CNN [35] model which will be the starting point for two-stage methods. Two-stage models as a first stage extract and propose regions of the input image as regions containing objects then as a second stage a CNN model is used to extract the features and finally a classification model is used to classify the regions. There are a multitude of methods offering the possibility of generating proposition regions. The most popular region generation algorithms are Selective Search (SS), Edge Boxes [36], Sliding Window. The R-CNN model therefore used the Selective Search (SS) algorithm to generate approximately 2000 regions then use AlexNet [14] to extract the features and finally the SVM algorithm to classify the regions. The propositions are readjusted to better fit the objects using linear regression. The biggest handicap of the R-CNN [35] model against these competitors at the time was its slowness. Ross Girshick et al. proposed an improved version of R-CNN called Fast R-CNN [37] much faster (0.5 frames per second) even if compared to the others, it was very slow, its power was its performance accuracy. To improve the speed of R-CNN, they used Region of Interest (RoI) pooling layer to share the computation across all the proposals instead of let the extraction module compute each of the proposed regions independently and used a single network to train and compute the extractor of features, classifier and proposals adjustment module. R-CNN [35] and Fast R-CNN [37] both used region proposal algorithm Selective Search in their model. SS being a completely independent algorithm that cannot be improved by training and being identified as a module taking the majority of the time of the entire detection process, Shaoqing Ren et al. in [4] have proposed to the Region Proposal Network (RPN) to make the regional proposals. Selective Search algorithm is therefore replaced by RPN which is a module generating bounding boxes or regions with objectness scores for each image given as input. The advantages of using RPN is the generation of region proposal with high accuracy and the improvement of the speed of the overall process (17 frames per second). They then, thanks to RPN, develop Faster R-CNN [4] which is a fusion of RPN and other modules (classification, regression to readjust proposals). The region proposal generation module RPN can be independently trained and then use its proposals to train the other modules of the Faster R-CNN model or we can train it with the other modules as a single network in order to reduce the time of training from 25% to 50%. NMS or Soft-NMS [30] is used to prevent multiple proposals referring to a single object.

E. Data Augmentation

It is known that the larger the training data set is, the better the machine learning models are [26]. Data augmentation is a strategy that allows us to increase our training data with manipulation techniques such as changing colors, brightness, contrast, removing pixels, reversing the image, adding natural phenomena such as rain, snow, lightning, etc... The techniques of data augmentation are used to allow the models to be trained on datasets sufficiently provided with the different possible cases that the model will meet. These techniques are used for the most part to increase the training set. The techniques used to augment the images must reflect the reality of the data. We cannot, for example, simulate snow in an African savannah. The techniques of Data augmentation are diverse and can notably intervene in audio and video with the generation of audio files, the addition of background noise or modification of textures.

With the application of data augmentation to several deep learning problems, it has also been noticed that apart from generalizing the models, it also reduces the risks of overfitting.

IV. EXPERIMENTS AND RESULTS

The dataset we have is smaller than the Snapshot Serengeti dataset [5] because not all of the images have been annotated. Gold Standard Snapshot Serengeti Bounding Box Coordinates [7] is a dataset which contains 4010 images labelled by experts in the field and which have been annotated by Stefan Schneider et al.. This dataset contains 46 animal species.

Even if the dataset is very unbalanced in terms of the number of each species present, we decided to keep all the species in order to reflect reality as much as possible but other solutions [10], [11], [38] have ignored and therefore used a dataset containing only the most present species in the Snapshot Serengeti dataset.

To be able to train and test this dataset, the annotations which have been saved in a csv file have been converted to PASCAL VOC format [32]. The dataset was divided into two sub datasets, 80% for the training dataset (3208) and 20% for the dataset test (802).

To have a fairly complex and complete image base to hope to have good results, data augmentation was used on the training dataset images with their bounding boxes. Eight data augmentation techniques were applied to the images. The augmented training dataset contains 28872 images in final. The transformations applied to original images are presented below in Fig. 1.

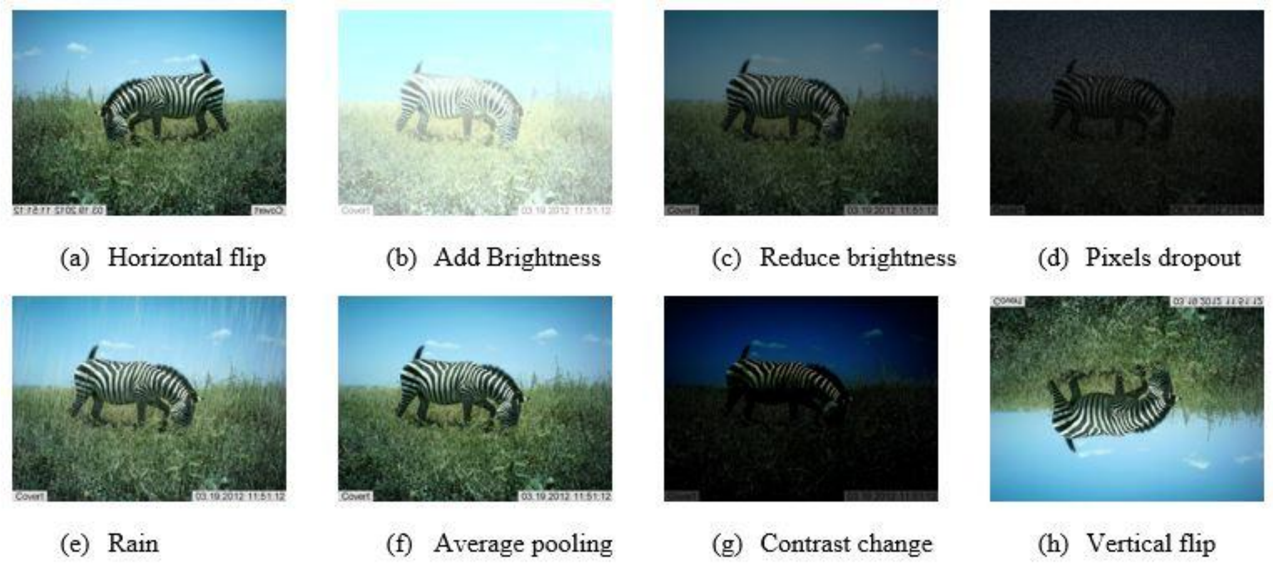


Fig. 1: The different data augmentation techniques applied to images in original training set

The original training dataset contains 3208 images with 16875 bounding boxes and the augmented training dataset contains 28872 images with 139539 bounding boxes. Table 1 and table 2 show the total number of occurrence for each species in the datasets so we will see the unbalanced nature of the datasets and the repartition of species in each dataset.

Table- I: Total number of occurrences for each species in the non-augmented (original) training set

Class Name	Number of occurrences	Percentage of class
Wildebeest	9604	0.5691259
Zebra	3290	0.1949630
Buffalo	943	0.0558815
Gazellethomsons	929	0.0550519
Impala	437	0.0258963
Hartebeest	345	0.0204444
Guineafowl	167	0.0098963
Gazellegrants	163	0.0096593
Warthog	135	0.0080000
Giraffe	113	0.0066963
Elephant	104	0.0061630

Cattleegret	86	0.0050963
Otherbird	69	0.0040889
Human	59	0.0034963
Stork	58	0.0034370
Hyenaspotted	44	0.0026074
Eland	44	0.0026074
Oxpecker	39	0.0023111
Baboon	33	0.0019556
Hippopotamus	32	0.0018963
Lion	30	0.0017778
Buffcrestedbustard	27	0.0016000
Topi	24	0.0014222
Reedbuck	22	0.0013037
Mongoose	11	0.0006519
Koribustard	9	0.0005333
Cheetah	7	0.0004148
Superbstarling	6	0.0003556
Porcupine	6	0.0003556
Dikdik	5	0.0002963
Serval	4	0.0002370
Bushbuck	4	0.0002370
Secretarybird	4	0.0002370
Jackal	3	0.0001778
Leopard	3	0.0001778
Rodent	3	0.0001778
Wattledstarling	3	0.0001778
Aardwolf	2	0.0001185
Ostrich	2	0.0001185
Aardvark	2	0.0001185
Greybackedfiscal	1	0.0000593
Vervetmonkey	1	0.0000593
Whiteheadbuffaloweaver	1	0.0000593
Rhinoceros	1	0.0000593

Table I: Total number of occurrences for each species in the augmented training set

Class Name	Number of occurrences	Percentage of class
Wildebeest	81973	0.5874558
Zebra	26494	0.1898681
Buffalo	7038	0.0504375
Gazellethomsons	6874	0.0492622
Impala	3871	0.0277413
Hartebeest	2518	0.0180451
Guineafowl	1325	0.0094956
Gazellegrants	1280	0.0091731
Warthog	1182	0.0084708
Elephant	908	0.0065071
Giraffe	895	0.0064140
Cattleegret	558	0.0039989
Otherbird	535	0.0038341
Human	495	0.0035474
Stork	452	0.0032392
Hyenaspotted	429	0.0030744
Eland	337	0.0024151
Oxpecker	328	0.0023506
Reedbuck	325	0.0023291
Baboon	248	0.0017773
Hippopotamus	238	0.0017056
Lion	236	0.0016913

Buffcrestedbustard	189	0.0013545
Topi	174	0.0012470
Porcupine	77	0.0005518
Mongoose	76	0.0005447
Koribustard	70	0.0005017
Dikdik	57	0.0004085
Superbstarling	45	0.0003225
Serval	41	0.0002938
Cheetah	41	0.0002938
Aardvark	29	0.0002078
Leopard	27	0.0001935
Bushbuck	23	0.0001648
Secretarybird	23	0.0001648
Jackal	19	0.0001362
Rodent	18	0.0001290
Wattledstarling	18	0.0001290
Aardwolf	15	0.0001075
Ostrich	14	0.0001003
Greybackedfiscal	9	0.0000645
Hare	7	0.0000502
Whiteheadbuffaloweaver	7	0.0000502
Rhinoceros	7	0.0000502
Waterbuck	7	0.0000502
Vervetmonkey	7	0.0000502

Note also that the datasets contain many obstructed, truncated, obscured and very small objects often in crowds and sometimes exposed to a strong light of a camera trap flash.

For our experiments, we used the Faster R-CNN [4] object detection model. The ResNet-101 deep neural network [17] has been used as a feature extractor given the excellent results reported by previous work.

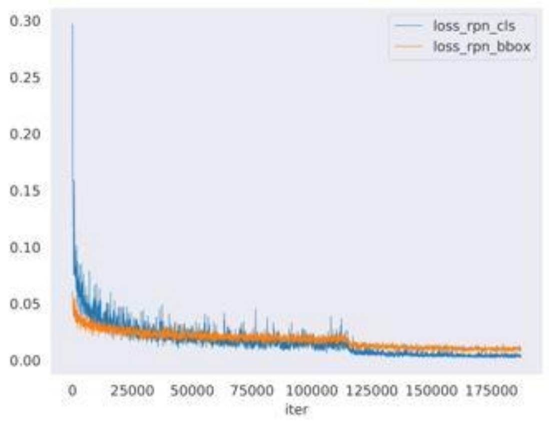
To show how using transformations on the original images dataset to make the data more complex and more consistent makes the trained model more efficient. We trained the same model with the same parameters on the original training set and the increased one and tested the models obtained for each training dataset on the same test set. The metrics used to show the performance differences are the model accuracy, recall and average precision (AP) on each class and the mean average precision (mAP).

The Gold Standard Snapshot Serengeti Bounding Box Coordinates dataset [7] having a structure that does not allow us to be able to use it as it is, it has been converted to the Pascal VOC format. The csv files containing the bounding boxes were used to generate the XML files (one file per image) containing the bounding boxes of each image. the XML files have been placed in the ‘Annotations’ folder. The images were placed in the ‘JPEGImages’ folder and the sets as text files were placed in the ‘ImageSets’ folder. After the data augmentation, the generated images were simply added to the ‘JPEGImages’ folder and the files containing their annotations were added to the ‘Annotations’ folder. A set file was created for the augmented images in order to train the model with and placed in the ‘ImageSets’.

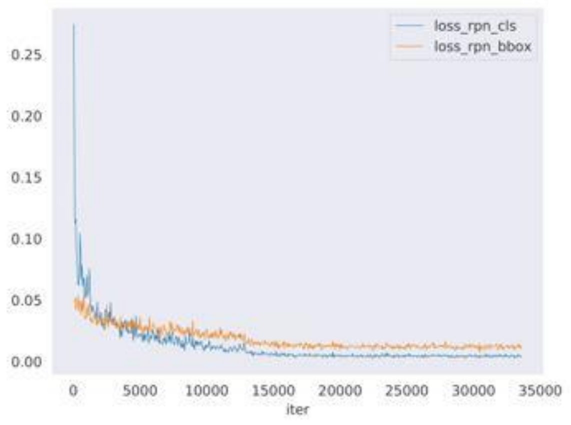
As said above, ResNet-101 [17] was used as the backbone for this model. It was pre-trained on the ImageNet dataset [39]. The Faster R-CNN model was trained using the technique named Approximate joint training. The optimization algorithm used is the Stochastic Gradient Descent (SGD) with a learning rate of 0.01, a momentum of 0.9 and a weight decay of 0.0001.

The model trained on the original dataset reached an accuracy of 97.79% with a mAP of 28.4% and an average of recall at 45% and the model trained on the augmented dataset reached an accuracy of 98.26% with a mAP of 95.3% and an average of recall at 96.53% which means that there are far fewer false positives and false negatives detected by the second model while also detecting the majority of the objects present in the images, which therefore makes it better than the first model.

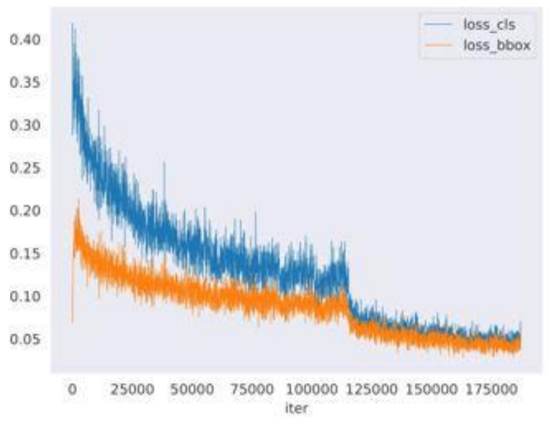
Fig. 2 shows the losses of the models during the training of the model on the different datasets. During the training of the model, several metrics were observed to see the behavior of the model. These metrics are rpn_cls_loss which penalize incorrect foreground/background prediction, rpn_box_loss which penalize inaccurate anchor boxes, cls_loss which penalize incorrect classification prediction and box_loss which penalize inaccurate proposals.



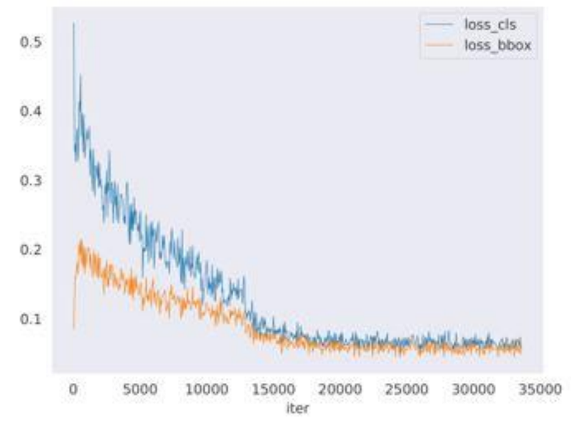
(a) **RPN** losses for augmented dataset



(b) **RPN** losses for non-augmented dataset



(c) **Fast R-CNN** losses for augmented dataset



(d) **Fast R-CNN** losses for non-augmented dataset

Fig. 2: Faster R-CNN losses of its two modules (RPN & Fast R-CNN) on the different training dataset (the original and the augmented)

We see that rpn_cls_loss and rpn_bbox_loss of the two trainings have the same behavior. They quickly stagnated because the weights of the features extraction network (Resnet-101) were initialized using pre-trained weights on ImageNet dataset. So RPN didn't have to do much to predict whether an image is a background or a foreground.

Unlike the previous losses, cls_loss and bbox_loss took a long time to converge and stagnate. They exhibit the same behaviors. This is explained by the fact that the R-CNN took more time to learn since new classes are in play

Tables 3 and 4 respectively contain the evaluations of the model obtained with the original dataset and the augmented dataset for each class.

Table II: Evaluation results of the model trained with the non-augmented training set on the test set

Class	gts	dets	Recall	Ap
Aardvark	2	0	0	0
Aardwolf	0	0	0	0
Baboon	2	91	0.5	0.045
Buffalo	242	687	0.426	0.251
Buffcrestedbustard	1	50	0	0
Bushbuck	1	0	0	0
Cattleegret	0	12	0	0
Cheetah	0	4	0	0
Dikdik	2	0	0	0

Eland	4	99	0.75	0.035
Elephant	32	218	0.594	0.294
Gazellegrants	34	137	0.706	0.406
Gazellethomsons	48	272	0.896	0.610
Giraffe	28	131	0.786	0.649
Greybackedfiscal	0	0	0	0
Guineafowl	33	61	0.667	0.632
Hare	1	0	0	0
Hartebeest	11	206	0.727	0.379
Hippopotamus	4	66	1.0	0.518
Human	15	87	1.0	0.628
Hyenaspotted	20	79	0.7	0.541
Impala	106	367	0.868	0.696
Jackal	0	0	0	0
Koribustard	2	0	0	0
Leopard	1	0	0	0
Lion	7	86	0.286	0.109
Mongoose	0	26	0	0
Ostrich	0	0	0	0
Otherbird	12	44	0.583	0.297
Oxpecker	5	19	0.4	0.313
Porcupine	4	0	0	0
Reedbuck	24	81	0.5	0.377
Rhinoceros	0	0	0	0
Rodent	0	0	0	0
Secretarybird	0	0	0	0
Serval	4	0	0	0
Stork	5	16	0.6	0.182
Superbstarling	3	0	0	0
Topi	0	67	0	0
Vervetmonkey	0	0	0	0
Warthog	33	149	0.788	0.675
Waterbuck	1	0	0	0
Wattledstarling	0	0	0	0
Whiteheadbuffalowe aver	0	0	0	0
Wildebeest	2616	5737	0.81	0.706
Zebra	625	1645	0.813	0.756
mAP				**0.284**

Table III: Evaluation results of the model trained with the augmented training set on the test set

Class	gts	dets	Recall	Ap
Aardvark	2	5	1.0	1.0
Aardwolf	0	6	0	0
Baboon	2	11	1.0	1.0
Buffalo	242	352	0.992	0.909
Buffcrestedbustard	1	8	1.0	1.0
Bushbuck	1	7	1.0	1.0
Cattleegret	0	4	0	0

Cheetah	0	13	0	0
Dikdik	2	11	1.0	1.0
Eland	4	9	1.0	1.0
Elephant	32	73	0.969	0.909
Gazellegrants	34	61	1.0	1.0
Gazellethomsons	48	119	0.979	0.903
Giraffe	28	54	1.0	0.997
Greybackedfiscal	0	1	0	0
Guineafowl	33	39	1.0	1.0
Hare	1	0	0	0
Hartebeest	11	43	1.0	1.0
Hippopotamus	4	11	1.0	1.0
Human	15	32	1.0	1.0
Hyenaspotted	20	36	1.0	0.996
Impala	106	160	1.0	0.999
Jackal	0	6	0	0
Koribustard	2	8	1.0	1.0
Leopard	1	7	1.0	1.0
Lion	7	18	1.0	1.0
Mongoose	0	1	0	0
Ostrich	0	1	0	0
Otherbird	12	30	1.0	1.0
Oxpecker	5	11	1.0	1.0
Porcupine	4	7	1.0	1.0
Reedbuck	24	46	1.0	0.987
Rhinoceros	0	0	0	0
Rodent	0	0	0	0
Secretarybird	0	3	0	0
Serval	4	12	1.0	1.0
Stork	5	10	1.0	1.0
Superbstarling	3	6	1.0	1.0
Topi	0	12	0	0
Vervetmonkey	0	0	0	0
Warthog	33	59	1.0	1.0
Waterbuck	1	1	1.0	1.0
Wattledstarling	0	1	0	0
Whiteheadbuffaloweaver	0	0	0	0
Wildebeest	2616	4167	0.974	0.907
Zebra	625	912	0.976	0.909
mAP				**0.953**

To measure the accuracy of counting the species present on the images, the results obtained with our models are compared with the original data. Note that we have considered predictions with or more than 80% confidence. The number of each species is considered and with a single false positive in our predictions, the counting of the species in the image is considered to be false therefore our system gives a very strict accuracy since only the counting predictions exactly equal to the counting of original data is considered fair. Table 5 shows the counting accuracy for each model.

Table IV: Counting accuracy of each model on the test set

	Original model	Augmented model
False prediction	529	164
True prediction	273	638
Accuracy	34.03%	79.55%

In Fig. 3, the results of some counting predictions made by the model trained on the augmented dataset and in Fig. 4, the same images with bad counting predictions made by the model trained with the original dataset.

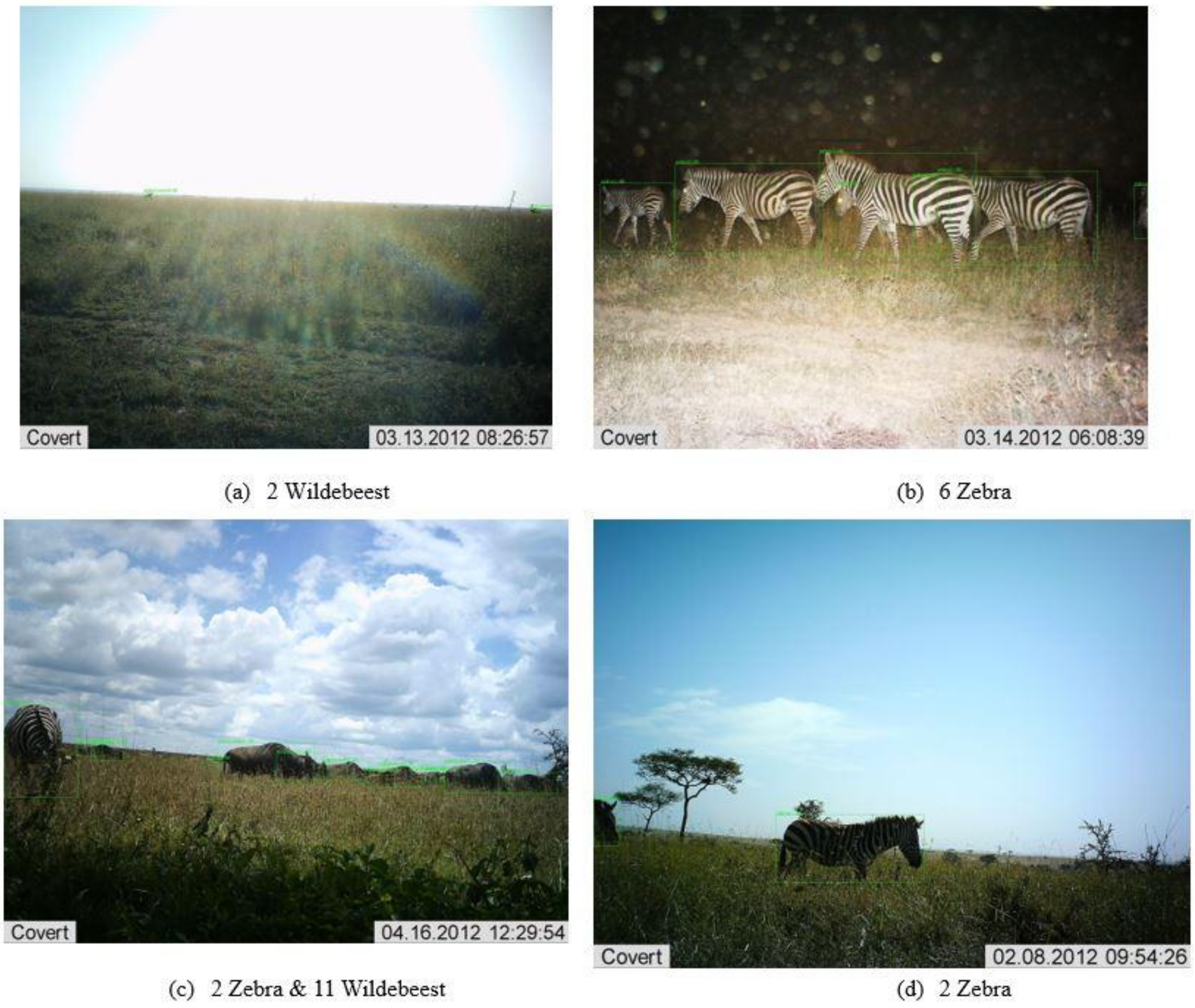


Fig. 3: 4 correct counting predictions made with the model trained on the augmented dataset

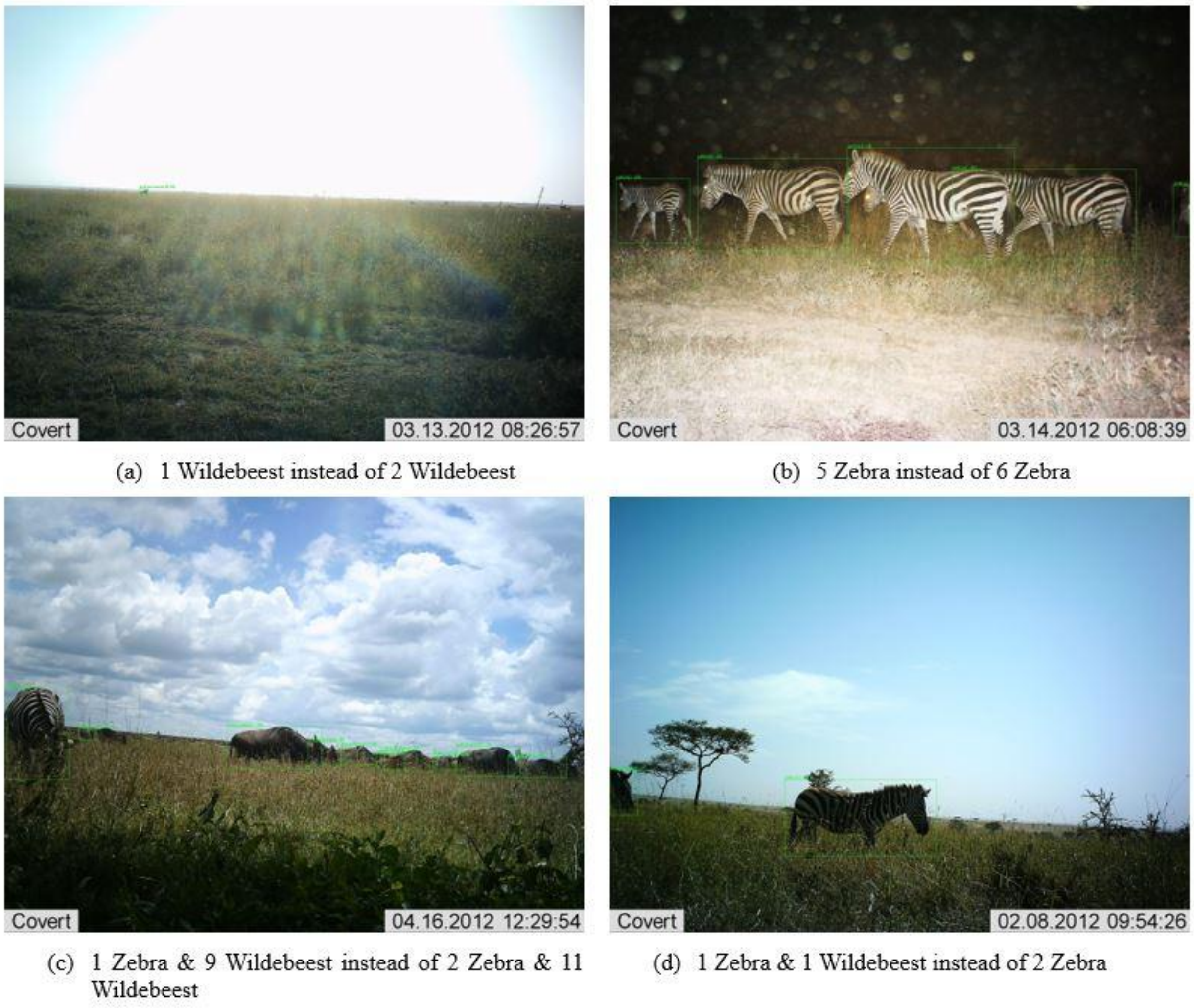

Covert
03.13.2012 08:26:57
(a) 1 Wildebeest instead of 2 Wildebeest
Covert
03.14.2012 06:08:39
(b) 5 Zebra instead of 6 Zebra
Covert
04.16.2012 12:29:54
(c) 1 Zebra & 9 Wildebeest instead of 2 Zebra & 11 Wildebeest
Covert
02.08.2012 09:54:26
(d) 1 Zebra & 1 Wildebeest instead of 2 Zebra

Fig. 4: 4 wrong counting predictions made with the model trained on the non-augmented dataset

The model trained on the original dataset shows by its performance that it has trouble detecting species with very few occurrences in the dataset. It has a really bad performance. Based on the recalls, we can say that it failed to detect 50% of the ground-truth bounding boxes and on top of that, it generates a lot of false positives. On the other hand, the model trained on the augmented dataset shows a satisfactory performance. It was able to detect up to 96.53% of ground-truth bounding boxes and managed to generate fewer false positives.

V. DISCUSSIONS

In this article, we investigated whether using data augmentation techniques on the original dataset containing images of animals captured by camera traps improves the detection performance of animal species in these images using an object detection model (Faster R-CNN [4]). We have reached an accuracy of 98.26% and a mAP of 95.3% with data augmentation against an accuracy of 97.79% and a mAP of 28.4% with the original dataset. Table 3 and table 4 clearly shows the difference in performance of the two models on the different species.

There are various possible explanations for these results. Data augmentation is one of techniques which are used to improve performance of computer vision task. In computer vision, get more data will almost always help [26]. When the dataset is small, we have relatively few examples and as result, we can have some mistakes for our new examples. Note that there are species that are largely underrepresented, which means that we cannot have enough examples to classify them correctly using the original dataset. Looking at the results presented in table 3 and 4, we notice that the species with the few occurrences have lower performances than the species with enough occurrences. Data augmentation helps to not overfit models and in addition make our model flexible enough to recognize various species in various conditions. Data augmentation refines the training of neural networks and allows them to make more efficient predictions and helps to balance datasets [22].

A lot of researches [9], [10], [11], [12] in this area has used classification models such as AlexNet [14], VGG [15], GoogLeNet [16], and ResNets [17] in order to classify images into a species class. The problem with these methods is that in most camera trap images, there are several animals and sometimes even several species. Even if certain methods here have performed well in the task of classifying certain images, they do not really reflect the real need of researchers because they do not perform well for tasks like counting and describing animals present in the images.

Others researchers have used object detection techniques such as Schneider et al. who compared the performance of two object detection models (Faster R-CNN [4] and YOLOv2 [23]) and found that Faster R-CNN is better than YOLOv2 for detecting and classifying animals in the images. Norouzzadeh et al. [13] as for them used a multistage model based on deep learning models to detect, count and describe animals present in the images. They achieved an accuracy of 94.9% for the classification of the species and 63.1% for the counting of the animals present in the images. Even if these methods have achieved good results, our results are better for the classification and counting of animals.

Taken together, our findings and the findings of previous studies indicate that the object detection models and in particular Faster R-CNN model is more suitable for the treatment of this automation problem. These findings also indicate that with a well-balanced dataset with enough data and precise data augmentation techniques, an accurate model with very good performance for each task (classification, counting and description) can be built and deployed.

VI. CONCLUSION

In this article, we have studied the potential advantages of training our Faster R-CNN [4] model on augmented images of animals captured by camera traps using specific data augmentation techniques. An object detection model has been used because it has been shown that they are more suitable for this type of task given the information to be extracted. We used two datasets, the original Gold Standard Snapshot Serengeti Bounding Box Coordinates [7] and the augmented one, to train a Faster R-CNN model with the same parameters. The model trained on the augmented dataset showed performance above the other model. It has 98.26% accuracy for classification of regions in categories of species and 79.55% accuracy for the counting of species present in images against 97.79% regions classification accuracy and 34.03% species counting accuracy for the model train on the original dataset. Given the performance of the Faster R-CNN model trained on augmented dataset, it can notably be used in animal detection and automatic information retrieval systems in order to greatly facilitate the task of ecological researchers. In the future, the model will be trained to recover other important information such as actions (standing, eating or moving) and characteristics (young or not) of the animals on the captured images.

REFERENCES

1. A. F. O'Connell, J. D. Nichols, and K. U. Karanth, "Camera traps in animal ecology: Methods and analyses," Camera Traps Anim. Ecol. Methods Anal., pp. 1–271, 2011, doi: 10.1007/978-4-431-99495-4.
2. L. Liu et al., "Deep Learning for Generic Object Detection: A Survey," Int. J. Comput. Vis., vol. 128, no. 2, pp. 261–318, 2020, doi: 10.1007/s11263-019-01247-4.
3. Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object Detection with Deep Learning: A Review," IEEE Trans. Neural Networks Learn. Syst., vol. 30, no. 11, pp. 3212–3232, 2019, doi: 10.1109/TNNLS.2018.2876865.
4. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.1109/TPAMI.2016.2577031.
5. A. Swanson, M. Kosmala, C. Lintott, R. Simpson, A. Smith, and C. Packer, "Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna," Sci. Data, vol. 2, pp. 1–14, 2015, doi: 10.1038/sdata.2015.26.
6. LILA BC, "Snapshot Serengeti - LILA BC", March 16, 2020. Accessed on: May 30, 2020. [Online]. Available: http://lila.science/datasets/snapshot-serengeti
7. Schneider, Stefan; Kremer, Stefan; Taylor, Graham, 2018, "Gold Standard Snapshot Serengeti Bounding Box Coordinates", https://doi.org/10.5683/SP/TPB5ID, Scholars Portal Dataverse, V2
8. K. Chen et al., "MMDetection: Open MMLab Detection Toolbox and Benchmark," 2019
9. A. Gomez Villa, A. Salazar, and F. Vargas, "Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks," Ecol. Inform., vol. 41, pp. 24–32, 2017, doi: 10.1016/j.ecoinf.2017.07.004.
10. L. Guignard and N. Weinberger, "Animal identification from remote camera images," pp. 1–4, 2016.
11. H. Nguyen et al., "Animal recognition and identification with deep convolutional neural networks for automated wildlife monitoring," Proc. - 2017 Int. Conf. Data Sci. Adv. Anal. DSAA 2017, vol. 2018-January, no. Figure 1, pp. 40–49, 2017, doi: 10.1109/DSAA.2017.31.
12. M. A. Tabak et al., "Machine learning to classify animal species in camera trap images: Applications in ecology," Methods Ecol. Evol., vol. 10, no. 4, pp. 585–590, 2019, doi: 10.1111/2041-210X.13120.
13. M. S. Norouzzadeh et al., "Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning," Proc. Natl. Acad. Sci. U. S. A., vol. 115, no. 25, pp. E5716–E5725, 2018, doi: 10.1073/pnas.1719367115.
14. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84–90. doi:10.1145/3065386
15. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., pp. 1–14, 2015.
16. C. Szegedy et al., "Going deeper with convolutions," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 07-12-June-2015, pp. 1–9, 2015, doi: 10.1109/CVPR.2015.7298594.
17. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-December, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.
18. Lin M, Chen Q, Yan S (2013) Network in network. arXiv:1312.4400.
19. Caruana R. (1998) Multitask Learning. In: Thrun S., Pratt L. (eds) Learning to Learn. Springer, Boston, MA
20. H. Yousif, J. Yuan, R. Kays, and Z. He, "Fast human-animal detection from highly cluttered camera-trap images using joint background modeling and deep learning classification," Proc. - IEEE Int. Symp. Circuits Syst., 2017, doi: 10.1109/ISCAS.2017.8050762.
21. S. Schneider, G. W. Taylor, and S. Kremer, "Deep learning object detection methods for ecological camera trap data," Proc. - 2018 15th Conf. Comput. Robot Vision, CRV 2018, pp. 321–328, 2018, doi: 10.1109/CRV.2018.00052.
22. A. Ahmed, H. Yousif, R. Kays, and Z. He, "Semantic region of interest and species classification in the deep neural network feature domain," Ecol. Inform., vol. 52, no. June 2018, pp. 57–68, 2019, doi: 10.1016/j.ecoinf.2019.05.006.
23. J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6517-6525, doi: 10.1109/CVPR.2017.690.
24. MacQueen, J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. 1. University of California Press. pp. 281–297.
25. Greig, D.M. & Porteous, B.T. & Seheult, Allan. (1989). Exact Maximum A Posteriori Estimation for Binary Images. Journal of the Royal Statistical Society, Series B. 51. 271–279. 10.1111/j.2517-6161.1989.tb01764.x.
26. T. Tiwari, T. Tiwari, and S. Tiwari, "How Artificial Intelligence, Machine Learning and Deep Learning are Radically Different?," Int. J. Adv. Res. Comput. Sci. Softw. Eng., vol. 8, no. 2, p. 1, 2018, doi: 10.23956/ijarcsse.v8i2.569.

27. McCulloch, W.S., Pitts, W. A logical calculus of the ideas immanent in nervous activity. Bulletin of Mathematical Biophysics 5, 115–133 (1943). https://doi.org/10.1007/BF02478259
28. Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), 2278–2324. doi:10.1109/5.726791
29. S. Dasiopoulou, V. Mezaris, I. Kompatsiaris, V. -. Papastathis and M. G. Strintzis, "Knowledge-assisted semantic video object detection," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 15, no. 10, pp. 1210-1224, Oct. 2005, doi: 10.1109/TCSVT.2005.854238.
30. N. Bodla, B. Singh, R. Chellappa and L. S. Davis, "Soft-NMS — Improving Object Detection with One Line of Code," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 5562-5570, doi: 10.1109/ICCV.2017.593.
31. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-December, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.
32. M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. IJCV, pages 303–338, 2010.
33. J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," 2018, [Online]. Available: http://arxiv.org/abs/1804.02767.
34. Liu, Wei & Anguelov, Dragomir & Erhan, Dumitru & Szegedy, Christian & Reed, Scott & Fu, Cheng-Yang & Berg, Alexander. (2016). SSD: Single Shot MultiBox Detector. 9905. 21-37. 10.1007/978-3-319-46448-0_2.
35. Girshick, Ross & Donahue, Jeff & Darrell, Trevor & Malik, Jitendra. (2013). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 10.1109/CVPR.2014.81.
36. Zitnick C.L., Dollár P. (2014) Edge Boxes: Locating Object Proposals from Edges. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8693. Springer, Cham
37. R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.
38. A. Loos, C. Weigel, and M. Koehler, "Towards automatic detection of animals in camera-trap images," Eur. Signal Process. Conf., vol. 2018-Septe, pp. 1805–1809, 2018, doi: 10.23919/EUSIPCO.2018.8553439.
39. Deng, J. et al., 2009. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255.

AUTHORS PROFILE

Panawè Touh obtained his bachelor in IAI-TOGO and is currently studying for a Master of Technology in the Department of Computer Science & Engineering of AP Goyal Shimla University where he is doing research in the fields of machine learning, computer vision and big data. He also participated in the development of several web and mobile applications. He is the author of two articles published in international jouinals.

Mukesh Sharma received the Master of Engineering degree in Computer Science and Engineering from Jaypee University of Information Technology in 2017. Currently, he is an Assistant Professor with the Alakh Prakash Goyal Shimla University, Himachal Pradesh. His research interest area is IoT, Machine Learning and Artificial Intelligence. His work has been published in International conference on I-SMAC.