Turk J Elec Eng & Comp Sci
() : –
© TÜBİTAK
doi:10.3906/elk-

Elimination of useless images from raw camera-trap data

Ulaş TEKELİ, Yalın BAŞTANLAR*
Computer Engineering Department, Izmir Institute of Technology, 35430, Izmir, Turkey

Received: .201 • **Accepted/Published Online:** .201 • **Final Version:** ..201

Abstract: Camera-traps are motion triggered cameras that are used to observe animals in nature. The number of images collected from camera-traps has increased significantly with the widening use of camera-traps thanks to advances in digital technology. A great workload is required for wild-life researchers to group and label these images. We propose a system to decrease the amount of time spent by the researchers by eliminating useless images from raw camera-trap data. These images are too bright, too dark, blurred or they contain no animals. To eliminate bright, dark and blurred images we employ techniques based on image histograms and Fast Fourier Transform. To eliminate the images without animals, we propose a system combining convolutional neural networks and background subtraction. We experimentally show that the proposed approach keeps 99% of photos with animals while eliminating more than 50% of photos without animals. We also present a software prototype that employs developed algorithms to eliminate useless images.

Key words: Camera-trap, Image Processing, Computer Vision, Object Detection, Background Subtraction, Convolutional Neural Networks, Deep Learning.

1. Introduction

Camera-traps are motion triggered cameras which are placed in the pathways of animals for the surveillance of the wild-life. Example camera-trap images are given in Figure 1. A properly working camera-trap may capture one thousand images in a month. Some photos of this large collection can be too dark, too bright, or blurred due to improper functioning of the camera. Also a considerable amount of captured photos do not contain any animals. Since researchers aim to observe animals, sort of images described above are considered as 'useless'. Examination of all the images gathered from a high number of cameras and deciding if there exists an animal in the image is a task that consumes a considerable amount of time for wild-life researchers.

Figure 1: Examples of images obtained from camera-traps.

*Correspondence: yalinbastanlar@iyte.edu.tr

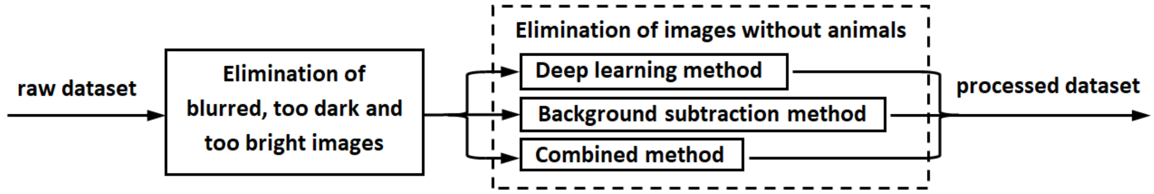


Figure 2: Pipeline of elimination process on raw camera-trap dataset.

The goal in our study is to automatically eliminate useless images from raw datasets of camera-traps, thus reduce the number of images to be visually examined by human experts. In our approach, blurred, too bright and too dark images are eliminated first (Figure 2). We propose novel approaches to discriminate blurred images from partially blurred ones and to discriminate too dark and too bright images from acceptable dark and bright ones. Next, the goal is to eliminate images without animals. In our work, two methods were evaluated to detect animals in camera-trap images, one is based on background subtraction (since camera-traps collect images with varying time intervals on the same scene) and the other one uses convolutional neural networks (CNN). We also investigated how to combine these two methods to obtain the best results. We achieved a higher animal/ non-animal image classification accuracy compared to the previous works. Moreover, we developed a software prototype that includes the image elimination modules mentioned above.

In Section 2, we summarize the related work in literature and explain our contributions in detail. We introduce our methods on eliminating blurred, too bright and too dark images in Section 3. Section 4 is devoted to describe the methods of object detection in order to eliminate images without animals. We present experiment results in Section 5 and give brief information about the prototype software in Section 6. Lastly, our conclusions are given in Section 7.

2. Related Work and Our Contributions

Studies on automatic animal detection and classification from images and videos taken in nature are relatively new. In [1], a video dataset was formed with sub-aqua cameras. Fish classification was performed on the regions obtained by separating moving objects from the background. A large feature set was used including color, shape, texture properties and moment invariants. A study dedicated to decrease the workload of wild-life researchers was conducted by Song and Xu [2]. In this work, birds were detected in videos and tracked with Kalman filter aiming to show the experts only the videos with a high probability of containing birds. With a similar goal, Weinstein [3] proposed a system where moving objects are detected from the videos that are captured in nature and the relevant frames are offered to the user. In [4], rather than camera-trap images, photographs from a museum database are used for species classification.

A species detection study on an actual camera-trap collection was first conducted by Yu *et al.* [5]. A dataset with 7000 images and 18 species was used. SIFT and LBP descriptors were combined into a feature vector and classified with SVM resulting in a classification accuracy of 82%. Chen *et al.* [6] was the first to use convolutional neural networks (CNN) to classify species from camera-trap images, using the dataset of University of Missouri that includes 20 species. Although the potential of CNN is a lot higher, because this study took place in 2014, 38% accuracy was obtained. In 2017, Gomez-Villa *et al.* [7] tested different CNN structures with a much bigger dataset (Snapshot Serengeti, 26 species, 780.000 images) and reported an accuracy of 60%. Another study on Serengeti dataset was made by Norouzzadeh *et al.* [8] where CNN models were trained from scratch. Classification accuracy increased up to 94% with the best model when the highest probability class is considered (top-1 accuracy). In this study, two class (animal and non-animal) image classification is also performed and 96.8% accuracy was reported for the best performing model. Another study [9] which used a different but again large camera-trap dataset reported 90.4% accuracy for species classification and 96.6% accuracy for animal/non-animal classification.

We do not aim species classification in this work, however our results for eliminating images without animals can be compared with the animal/non-animal classification results in literature. Previous studies that obtained 96% accuracy on this task [8, 9] were held with large and mixed collections of camera-trap images. Also those collections were cleaned such that no unusable (too blurred, too bright etc.) photos or unrecognizable animals remain. Our dataset is more challenging in the way that we handle raw image folders (a folder per camera-trap) and we do not mix train and test folders which suits to the real-life scenario where test images come from new camera-trap locations. Under these realistic conditions, a state-of-the-art image classification CNN (ResNet) reached only 80.7% accuracy on animal/non-animal classification. Our first contribution is that we increased it to 90.2% by training an object detector CNN (Faster R-CNN) to find animals and eliminate the images without any detected animals. We also investigated to use of suggested practices such as transfer learning, data augmentation and ensemble of networks to obtain best results.

Our second contribution is that we adapt a background subtraction technique to eliminate camera-trap images without animals for the first time. We also propose an approach combining CNN and background subtraction methods together. In our experiments, this combined method achieved 99.1% rate of keeping photos with animals while eliminating more than 50% of photos without animals.

Third, we have used the common technique of blur detection in frequency domain with a novel strategy of dividing the processed image into sub-images. This produced considerably better results in discriminating usable partially blurred images from completely blurred ones.

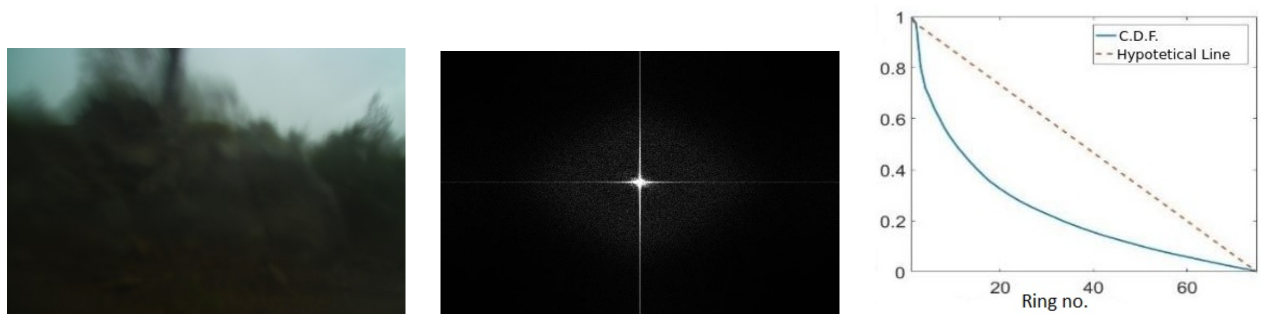
Our novelty is to discover the best-performing combinations of the related methods and to integrate them for the real world problem of eliminating useless images raw camera-trap datasets. Thus, we are the first to present what could be expected from a complete system under realistic conditions. Moreover, we developed a software prototype including these elimination modules. There are a few data management software proposed to manage camera-trap folders and label images [10–12]. However, since no automatic elimination is performed, these software do not reduce the number of images to be visually checked by researchers.

3. Blurred, Bright and Dark Image Elimination

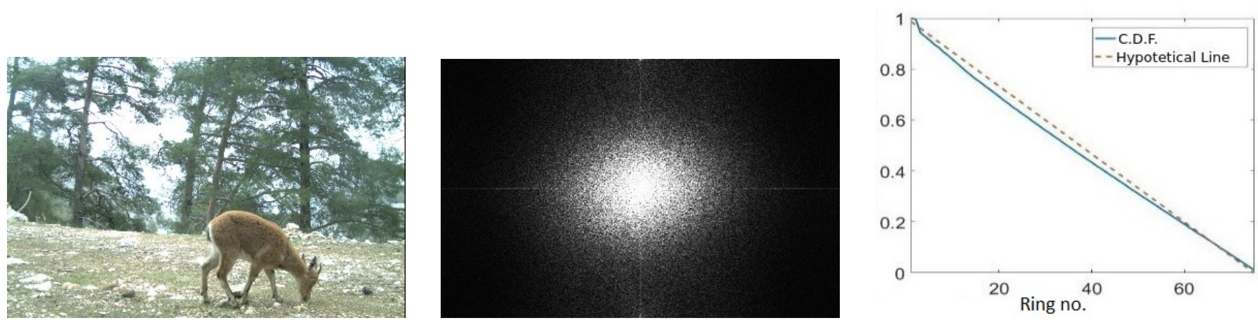
3.1. Blurred Image Elimination

Many approaches on blur detection were proposed in the last 25 years. Pavlovic and Tekalp [13] proposed a method that uses maximum likelihood on spatial space to detect blur. Narvekar and Karam [14] put together a cumulative probability metric, whereas Tong *et al.* [15] used wavelet transformation based on edge shapes and edge sharpness. Fourier Transform is another method used commonly on blur detection. Low frequency coefficients are represented close to the center of the centered spectrum which is obtained by Fourier Transform. Since the intensity differences between neighboring pixels of a blurred image is too low, a blurred image must produce a spectrum with very low frequencies (accumulation in the center). Figure 3 shows two images that are labeled as blurred and clear and their corresponding Fourier spectra.

Dosselman and Yang [16] place rings with varying radii on the Fourier spectrum's center and calculate the responsiveness of areas between rings. The sum of pixels values between each ring is recorded and used to form a cumulative distribution function (CDF) shown in Figure 3 (last column). The number of rings in this example is 75. For each ring, values from the outermost ring up to that ring are summed up and divided by the total sum of 75 rings (i.e. all spectrum). That is why we reach CDF value of 1.0 when the ring number is 1. Also, a hypothetical line is shown in the figure, representing an image with equal frequency distribution. Detection of blur using CDF is as follows. For every ring, the hypothetical line's value for that ring is subtracted from the ring's CDF value. The results are summed up and divided to the summation of the hypothetical line's values. The obtained value is assigned as ϕ. The images with lower ϕ than a threshold are labeled as blur.



(a)

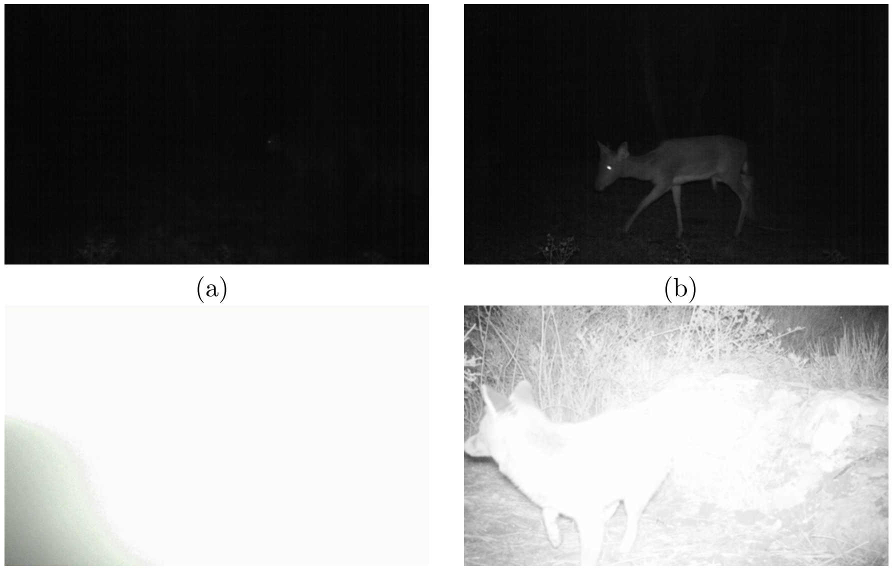


(b)

Figure 3: (a) From left-to-right: A blurred image, its Fourier transform and computed cumulative distribution function (CDF) with the algorithm given in [16]. (b) From left-to-right: A clear image, its Fourier transform and computed CDF.

Figure 4: Partially blurred images in raw camera-trap dataset

The algorithm described above [16] is very sensitive to blurriness and does not enable us to determine a threshold that will also identify partially blurred images. These images are the ones that contain clear parts. We do not want to eliminate these partially blurred images since animals can be identified in the clear regions.



(a) (b) (c) (d)

Figure 5: Examples of too dark (a), dark but useful (b), too bright (c) and bright but useful (d) images.

Examples of partially blurred images can be seen in Figure 4. We propose an approach based on [16] and compute blurriness in different parts of images. If only a few parts of the image are blurred, it is not eliminated. We divide the images into a fixed number of sub-images and for each sub-image we perform blur detection. The number of blurred sub-images is divided to the total number of sub-images to obtain the blur percentage of an image. In our experiments, we divided images into 16 equal sub-images and set the blur percentage threshold as 0.75, meaning if an image has 12 or more sub-images that is identified as blur, that image is labeled as blurred. For sub-images, the number of rings was decreased to 35 from 75 and the threshold for ϕ value was set to -0.03.

3.2. Bright and Dark Image Elimination

To eliminate too bright and too dark photos, a histogram based analysis is performed to estimate the darkness and brightness levels. To decrease the false negative results, partial dark and partial bright images are not specified as useless. Examples of too dark, too bright and useful (i.e. acceptable) images can be seen in Figure 5. Equation 1 shows dark pixel ratio (p_d) and bright pixel ratio (p_b) where $hist(i)$ denotes the number of pixels with intensity value i. Ratios are in [0,1] range. We observed that taking the square is more effective since it trivializes small values. These equations assume pixels with intensity value ≤ 20 are dark and pixels with intensity value ≥ 220 are bright, where intensity range is [0,255]. These are empirical values based on our observations on the dataset. We tested the proposed formula for threshold values from 128 to 255 for bright images and from 0 to 128 for dark images and chose the best performing values. Thresholds on p_d and p_b were set as 0.94 and 0.84 respectively. Again these threshold values are outcomes of an exhaustive search where target range is [0.5,1] for both bright and dark image testing. Images, whose darkness or brightness is higher than these thresholds, are eliminated.

$$p_d = \left(\frac{\sum_{i=0}^{20} hist(i)}{\sum_{i=0}^{255} hist(i)}\right)^2 \qquad p_b = \left(\frac{\sum_{i=220}^{255} hist(i)}{\sum_{i=0}^{255} hist(i)}\right)^2 \tag{1}$$

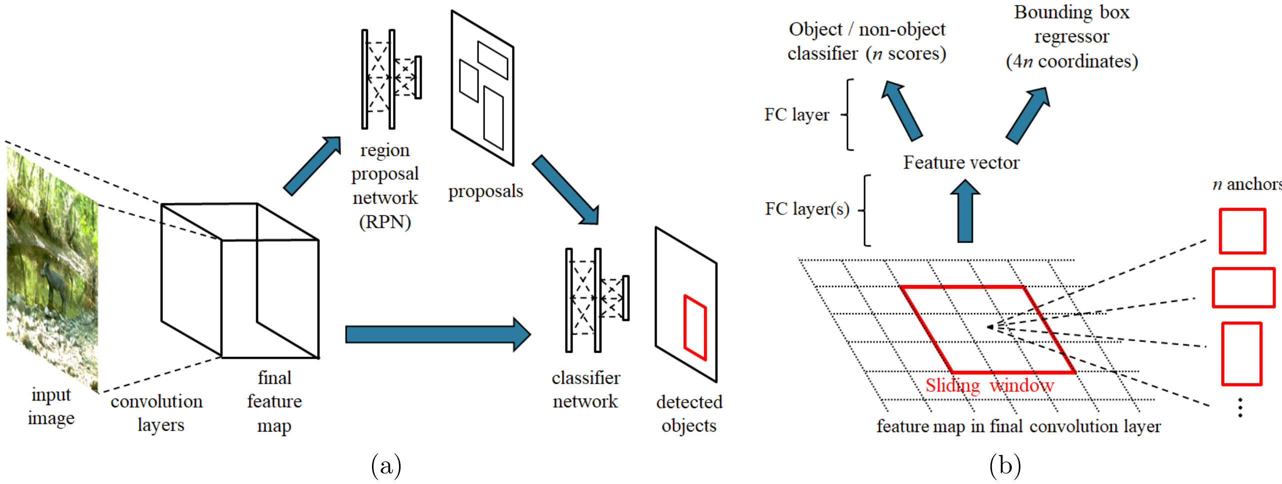


Figure 6: (a) Faster R-CNN [20] comprises two separate networks, Region Proposal Network (RPN) and Classifier Network, sharing the same backbone CNN. (b) RPN uses sliding window approach on the last convolutional layer to produce region proposals.

4. Detecting Images with Animals

4.1. Animal Detection with Deep Learning

Convolutional Neural Networks (CNN), especially after AlexNet [17] won the ILSVRC [18] competition of image classification in 2012, have been effectively used on many tasks of computer vision, including object detection. There are quite a few CNN approaches developed for object detection. Let us quickly review some of those.

OverFeat [19] is one of the earliest ones. In OverFeat, to detect object location, CNN with a classifier and a regressor head is trained. Objects are searched in the image in a sliding window fashion. In Faster R-CNN [20], object proposals are made through a Region Proposal Network (RPN) which shares last convolutional layer of CNN with a classifier network (Figure 6a). Then the proposed regions are classified.

YOLO [21] and SSD [22] use a different approach to process the image. They divide the image into regions and train a single neural network that predicts bounding boxes and class probabilities for each region. With this increased speed, recently, YOLO and SSD reached the detection accuracy of Faster R-CNN while processing real-time.

We chose Faster R-CNN for our object detection module. Main reasons of this choice are its proven effectiveness on different datasets and the abundance of documentation and source codes. As mentioned above, Faster R-CNN consists of two separate networks, sharing the same backbone CNN which extracts features. RPN uses sliding window approach on the last convolutional layer of the backbone CNN and at each position it determines 9 anchor boxes (3 different scales and 3 different sizes). For each anchor box, an objectness score is produced with a classifier head and 4 offset values are produced with a regressor head to make the proposal boxes more precise (Figure 6b). This usually totals up to 20000 anchor boxes with objectness scores for each image. Then a threshold is applied to eliminate low-score ones, and non-maximum suppression is used to eliminate overlapping boxes. To further decrease the number, top 300 anchor boxes with highest scores are selected to feed the classifier network. Classifier network classify these proposal regions using the corresponding areas on the last convolutional layer of the backbone CNN.

Since we do not aim species classification [8] or perfectly localizing an animal within the image [24], we kept an image if any animal is detected in it, and eliminated otherwise. Faster R-CNN is trained as a two-class classifier where all animals constitute the samples of positive training set. This approach is also a good choice for the situations where one can encounter animals which do not exist in the training set.

We also need to clarify why we employed an object detector instead of training an image classifier for animal/non-animal image classification. The reason is that general purpose image classifiers such as ResNet [23] does not perform well enough for our dataset obtained from Ministry of Forest and Water Affairs. As the details will be given in Section 5.2, we used separate cameras in training and test set which suits to the real-life scenario where test images come from new camera-trap locations. However, in studies in literature [6–8] same camera-traps are used for training and test, thus the same scenes exist in both training and test sets. The latter will be referred as mixed dataset. When a mixed dataset is used, an effective image classifier exploits background scene information to discriminate between animal and non-animal images. However, when new scenes come, its accuracy drops since it does not perform well for the scenes it did not see before. Table 1 shows the comparison of performance between state-of-the-art classification network ResNet [23] and Faster R-CNN [20] on separate and mixed versions of the same dataset. While ResNet accuracy drops significantly on separate dataset, drop in Faster R-CNN is limited since it is trained to find animals in images.

Table 1: ResNet [23] and Faster R-CNN [20] accuracies for animal/non-animal image classification.

	Faster R-CNN Accuracy	ResNet Accuracy
Mixed Dataset	94.3 %	95.6 %
Separate Dataset	90.2 %	80.7 %

4.2. Animal Detection with Background Subtraction

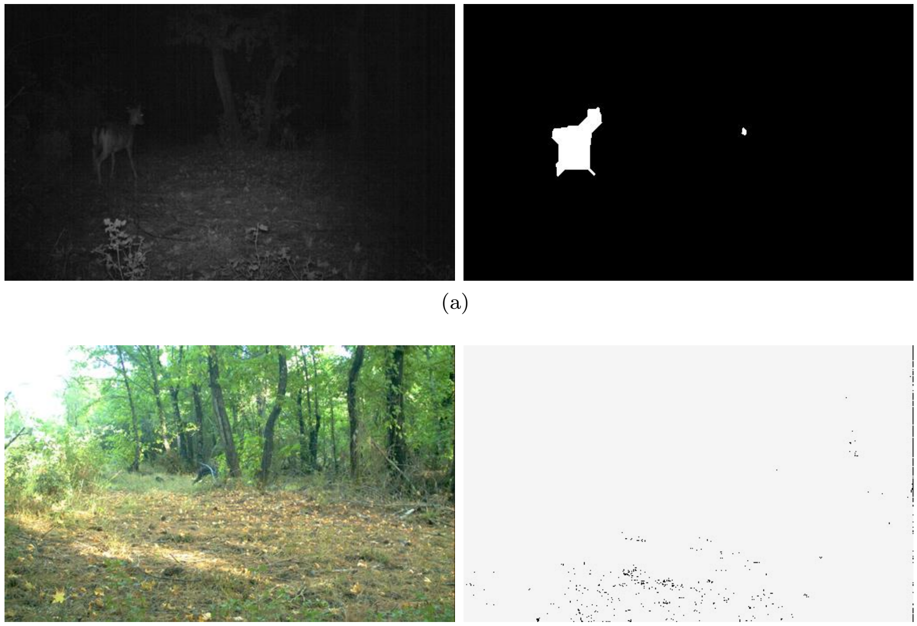
Background subtraction is a common approach to detect the moving objects in real-time videos. We decided to evaluate this approach since the camera-trap image sequences show strong resemblance to videos. Camera-traps collect images with varying time intervals on the same scene, resulting in a long image sequence with a single background.

A comprehensive review of background subtraction algorithms exists in [25]. We preferred to use Gaussian Mixture Model due to its compatibility with bi-modal backgrounds. In this method, each pixel is modeled by a mixture of K Gaussian distributions (K is a small number from 3 to 5). Different Gaussians are assumed to represent different colors. The probability that a pixel has a value of x can be written as

$$p(x) = \sum_{j=1}^{K} w_j \mathcal{N}(x, \theta_j) \tag{2}$$

where $\mathcal{N}(x, \theta_j)$ denotes the probability of x in the j^{th} Gaussian component which has parameters θ_j. Here, w_j is the weight parameter of j^{th} Gaussian component, representing the time proportion that color stays in the scene.

Static single-color objects tend to form tight clusters in the color space while moving ones form wider clusters due to different reflecting surfaces during the movement. Thus, w_k/σ_k is used as the fitness value to represent staying long and more tight. Higher fitness value refers to having higher probability to be a background component. The K distributions are ordered based on the fitness value and the first B distributions are used



(a)

(b)

Figure 7: A successful (a) and a failed (b) example of detecting animals with background subtraction. Images on the left are two consecutive images in raw camera-trap dataset. Since the difference between images is too much, the background subtraction result of second image implies the image has animal while there is not.

as a model of the background of the scene where B is estimated as

$$B = \underset{b}{\operatorname{argmin}}\left(\sum_{j=1}^{b} w_j > T\right) \tag{3}$$

where T is the threshold for the minimum acceptable fraction of the background model. If a pixel is more than 2.5σ away from any of B distributions, it is marked as a foreground pixel. To adapt to changes in illumination, an update scheme is applied such that every new pixel value is checked against existing model components in order of fitness. The first matched model component is updated. If no match is found, a new Gaussian component is added. For better adaptation to the scene, in [26], this method was improved in a way that not only the parameters but also the number of components of the mixture is constantly adapted for each pixel.

The images obtained from background subtraction goes through a series of morphological operations. After this, connected component analysis is applied to images to obtain the areas of the foreground objects. Objects whose area is higher than a threshold are defined as foreground objects. Figure 7a shows a successful example of a component defined as object.

Failures usually occur when lighting substantially changes between two consecutive images (an example is given in Figure 7b). Since camera-trap image sequence is collected from the same camera-trap during varying time intervals, there are cases where the time interval between two images is low but lighting substantially changes or where the time interval between two images is high but the lighting and background on these images looks identical (two images captured on same hours of different days). It is necessary to minimize the differences between frames to achieve good results. For this purpose, we propose an algorithm to group images with the same background.

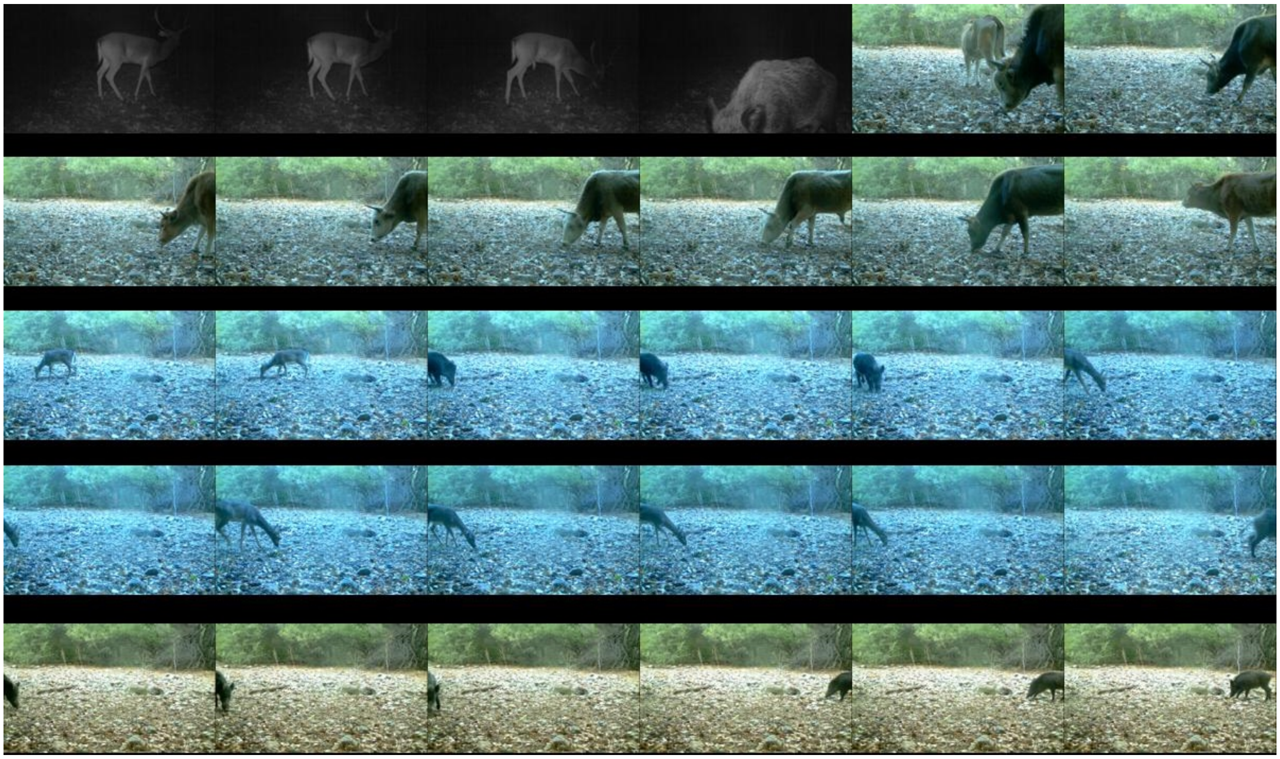


Figure 8: Clusters that show up after sorting the images. Starting from top-left, 1st, 5th, 13th, and 25th images are starting points of new clusters.

First, we create a similarity metric between two images, by comparing images pixel-by-pixel. If the absolute difference of a pixel between two images is higher than an empirical threshold, we count that pixel as 'changed'. The percentage of the changed pixels constitute our similarity metric. A low percentage indicates high similarity between two images. After we find the most similar image to the first image on the image series, we put it right after the first image in series and then we start to search the most similar image to the second image in series and so on. We also cluster sorted images from where the lighting changes drastically (low similarity between consecutive images) on image series. We observe that images captured at night usually grouped as one cluster while images captured during daytime usually clustered into several groups. An example clustering result can be seen in Figure 8. Later, we apply background subtraction to each cluster separately. To put it differently, the background model that is learned is forgotten before processing a new cluster. The flow of our background subtraction approach is given in Figure 9.

The proposed sorting algorithm improves the results since it decreases the number of the failures, especially the ones similar to Figure 7b. When the images are not sorted, a few consecutive images share the same background (illumination) and every substantial change in lighting results in a failure. However, after sorting, many more images benefit from the same background (such as images taken at night or images of the same time of the day but taken at different days). Thus, failed cases occur less often.

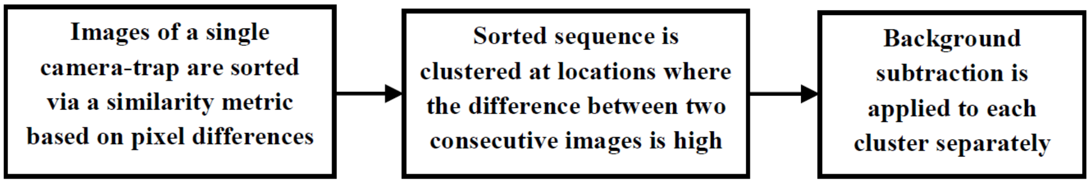


Figure 9: Steps of the proposed pre-processing for background subtraction approach.

Table 2: Blurred image classification results

Actual Classes	# of images	Proposed Approach			Original Approach [16]		
		Blurred	Clear	Accuracy	Blurred	Clear	Accuracy
Blurred	186	175	11	94.1%	182	4	97.8%
Clear	325	0	325	100%	1	324	99.9%
Partially Blurred	181	20	161	88.9%	60	121	66.8%
TOTAL	692			95.5%			90.6%

5. Experiments and Results

Our raw camera-trap dataset consists of nearly 40000 camera-trap images provided by the Ministry of Forest and Water Affairs, Republic of Turkey. These images had been collected from different cameras and mostly stored such that each folder contains images from a single camera (one background scene). Firstly, we manually scanned the images and labeled too dark, too bright and blurred ones. Next, we added bounding-box annotations on more than 2500 images with animals in Pascal VOC annotation format to be used for experiments of detecting images with animals. One thing we paid attention during the creation of annotated dataset is to ensure variation in terms of scenes, lighting conditions and animals. These images and their annotations are available on http://cvrg.iyte.edu.tr/.

5.1. Experiments on Eliminating Blurred, Too Dark and Too Bright Images

We prepared 692 images for blur detection experiments. 186 of them are blurred, 181 of them are partially blurred while the remaining 325 images are clear. Table 2 shows the classification results of both the original method [16] and proposed approach explained in Section 3.1. With the proposed approach, out of 186 blurred images, 175 are labeled as blurred, achieving 94.1% accuracy. This ratio of elimination is good since it saves human time. All of the clear images are remained, and for partially blurred images, only 20 out of 181 images are incorrectly labeled as blurred, achieving 88.9% accuracy. As seen in the table, this is much better than the 66.8% accuracy of original approach [16]. As a conclusion, the proposed approach of dividing the image into sub-images before processing eliminated two-thirds of incorrect blurred detections and it is important because these images will not be visually checked by experts in this scenario.

In Section 3.2, we explained our method of eliminating too dark and too bright images. We prepared 1017 too dark, 7 too bright (they are rare) and 2250 useful images for the experiments. Set of useful images contains many dark and bright images close to the borderline. The success of classification can be seen in Table 3. Only 11 dark images are incorrectly classified, no errors made on bright and useful images. Thus, this module is more effective than the blur elimination since it eliminates 99% of useless photos with no false-negatives.

Table 3: Confusion matrix for detection of too bright and too dark images

Classes	Predicted Classes		
	Dark	Bright	Useful
Dark	1006	0	11
Bright	0	7	0
Useful	0	0	2250

Table 4: Datasets used for the object detection experiments

Datasets		# of Train Images	# of Test Images
Ministry of FWA	DS-1	958	707
	DS-2		1248
Missouri University	DS-3	871	1474

5.2. Experiments on Eliminating Non-animal Images

We present the experiment results in three subsections. Results of deep learning methods are given in Section 5.2.1, results of background subtraction method are given in Section 5.2.2 and finally Section 5.2.3 presents the performance of combining these two methods.

All datasets used in Section 5.2 are shown in Table 4. In Ministry of FWA dataset, we have 958 images in our training set and 1955 images in our test set. The cameras in training and test sets are separate. We formed two separate test sets for Ministry of FWA images. One set (DS-1) contains low number of animals while the other test set (DS-2) has high number of animals (cf. Table 4). We observe that any camera-trap folder follows one of these two patterns and we aimed to analyze results separately.

In addition to Ministry of FWA dataset, we use a camera-trap dataset (DS-3) provided by University of Missouri [6]. We used DS-3 only for Section 5.2.1 since this dataset is not in raw folders and background subtraction method cannot be applied.

5.2.1. Experiments on Eliminating Non-animal Images with Deep Learning

As mentioned in Section 4 we trained a Faster R-CNN model. All animals are regarded as one class during CNN training in accordance with our goal of eliminating images without animals and remaining the ones that have animals regardless of their species.

During training we make use of transfer learning. We use pretrained weights on ImageNet dataset for the backbone architecture which is VGG16. We initialize weights of the fully connected layers of RPN and classifier network with zero mean and a standard deviation of 0.01. At test time, we keep an image if an animal is detected in it. Success of the system is measured with two criteria. One is the elimination rate of images without animals and the other is the remain rate of images with animals. We desire both rates to be high. Firstly, we trained and tested Faster R-CNN using Ministry of FWA images (DS-1 and DS-2 in Table 4). Results are shown in Figure 10 where eliminated image and remained image accuracies are depicted separately for different score thresholds. An increased threshold requires Faster R-CNN object boxes have higher confidence scores not to eliminate an image. It results in higher elimination accuracy but remained image accuracy drops since it starts to miss actual animals. On the left side (threshold$\leq$50) accuracies do not change since no Faster R-CNN object box has probability less than 0.5 (otherwise box would have been classified as background). Table 5 shows the detailed result of the experiment when threshold is kept at 0.5. On the average of two datasets, average of

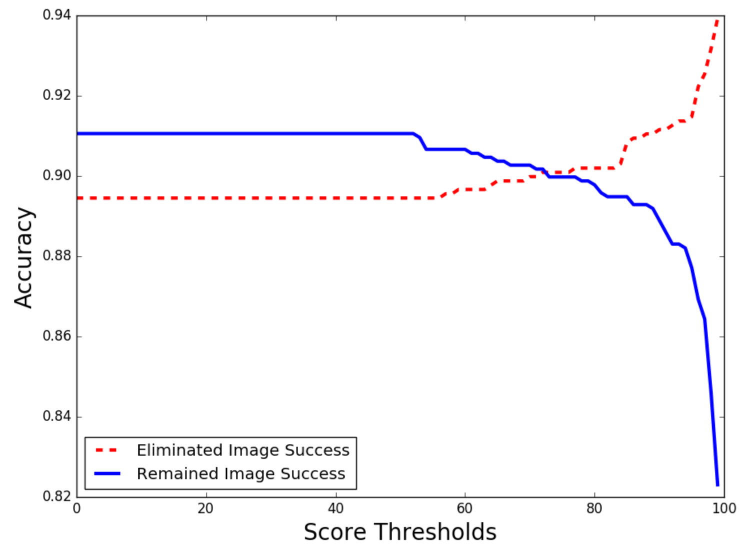


Figure 10: Results on deep learning experiments with different score thresholds.

Table 5: Percentages of eliminated and remained images with deep learning

Dataset	# of images		Success Rate		
	Animal	Empty	Eliminated	Remained	Accuracy
DS-1	76	631	90.8%	51.3%	86.4%
DS-2	941	307	86.9%	94.1%	92.3%
TOTAL	1015	938	89.5%	91.1%	90.2%

eliminated and remained image accuracies is 90.2%.

We also tested our model trained with Ministry of FWA on Missouri University test set (DS-3). The results are shown in Table 6. Accuracy shows a decline, pointing out that the generalization capacity of a model trained with a camera-trap dataset from a single source is limited. This result is in conformance with [27] where authors trained a CNN with Snapshot Serengeti dataset and tested the model with an 'out-of-distribution' dataset from Canada. In their study, the species classification accuracy decreased to 82% from 97%.

Another experiment we perform was to investigate the performance of ensemble of trained neural networks. Ensemble of NNs are quite popular with CNNs in different domains [28, 29]. For this purpose, we trained four separate networks to be used as the classifier of Faster R-CNN model (Figure 6) each use different and random 80% portions of the training set of Ministry of FWA. They share the same RPN. At test time, we ensemble them by unweighted averaging. In other words, for each window proposed by RPN, the scores of four classifiers are averaged.

Table 6: Percentages of eliminated and remained images on DS-3 with CNN trained with DS-1 & DS-2

Dataset	# of images		Success Rate		
	Animal	Empty	Eliminated	Remained	Accuracy
DS-3	886	588	68.3%	81.9%	76.4%

Table 7: Comparison between Ensemble of Networks and Baseline Learner using Ministry of FWA dataset

Methods	# of images		Success Rate		
	Animal	Empty	Eliminated	Remained	Accuracy
Ensemble of Networks	1015	938	89.5%	92.1%	90.7%
Baseline Learner			89.4%	91.0%	90.2%

(a)

(b)

(c)

(d)

Figure 11: Some examples of correct detections (a,b), missed animals (c) and false-positive detections (d) with deep learning method.

Test set consisting of both DS-1 and DS-2. Results are shown in Table 7 where baseline learner refers to the single Faster R-CNN that uses 100% of training data. When we compare baseline learner and ensemble of networks, we observe a small improvement in total accuracy as expected.

5.2.2. Experiments on Eliminating Non-animal Images with Background Subtraction

Although deep learning gives very good elimination and remained percentages (both around 90%), a few problems were noticed when we examined the false results. In addition to the successful detections (examples shown in Figures 11a and 11b), some animals were missed due to the similarity of their texture with the background (Figure 11c), whereas some large stones are mistaken as animals (Figure 11d). These problems can be fixed with background subtraction since it will detect animals that was not previously there and it will not detect rocks that stay in every frame.

As explained in Section 4.2, we sort and cluster images with the same background, we apply background

Table 8: Percentages of eliminated and remained images with background subtraction approach

Datasets	# of images		Success Rate		
	Animal	Empty	Eliminated	Remained	Accuracy
DS-1	76	631	60.6%	75%	62.0%
DS-2	941	307	46.9%	91.9%	80.8%
TOTAL	1017	938	56.1%	90.8%	74.0%

Table 9: Percentages of eliminated and remained images with combined method

Datasets	# of images		Success Rate		
	Animal	Empty	Eliminated	Remained	Accuracy
DS-1	76	631	60.0%	89.4%	63.1%
DS-2	941	307	43.3%	99.9%	85.9%
TOTAL	1017	938	54.5%	99.1%	77.6%

subtraction to each cluster of images on its own. The experiment results on DS-1 and DS-2 are given in Table 8. Our first observation is that, for DS-1, remained rate increased to 75% (cf. Table 5) catching most of the animals that are missed by deep learning method. On the other hand, the eliminated rate is lower than that of deep learning method (cf. Table 5). By examining mistakes, we observed that although some false-positives such as given in Figure 11d do not occur with background subtraction, other false-positives occur due to sudden changes of scene illumination. In total, the number of false-positives increase in background subtraction method, causing low eliminated rate. We also observed that remained images (true-positives) in both methods are partially different from each other which motivated us to perform experiments in Section 5.2.3.

5.2.3. Experiments on Eliminating Non-animal Images with Combined Method

With the observation that the two methods generally fail on different images (explained in Section 5.2.2), we designed an experiment where the decisions of both methods are combined. To eliminate an image, both methods must vote so. Otherwise, it is enough for either method to vote to remain an image in order to remain an image. This caused a drop on eliminated image accuracy and reduced it to 54.5% but the remained image accuracy reached %99.1. Table 9 shows the results of this experiment. When we examined the missed 0.9%, we noticed that missed animals also are seen in neighbor images that are remained (camera-traps keeps capturing until there is no movement in scene). Thus, we can say that around 500 images without animals were eliminated with no individual animal was missed.

6. Software

A prototype software was developed to be able to apply the proposed elimination algorithms on raw camera-trap data. Potential users of this software is wild-life researchers. A user is able to choose what type of images to be eliminated and which method to use for the elimination (when multiple methods available). Software tags images according to algorithm results and lets the user go through the images with selected tags. In addition, software contains must-features of any image management software such as choosing a folder or any number of images from a folder, manually tagging one or multiple images and filtering based on tags. A screenshot showing the graphical user interface of our prototype software can be seen in Figure 12.

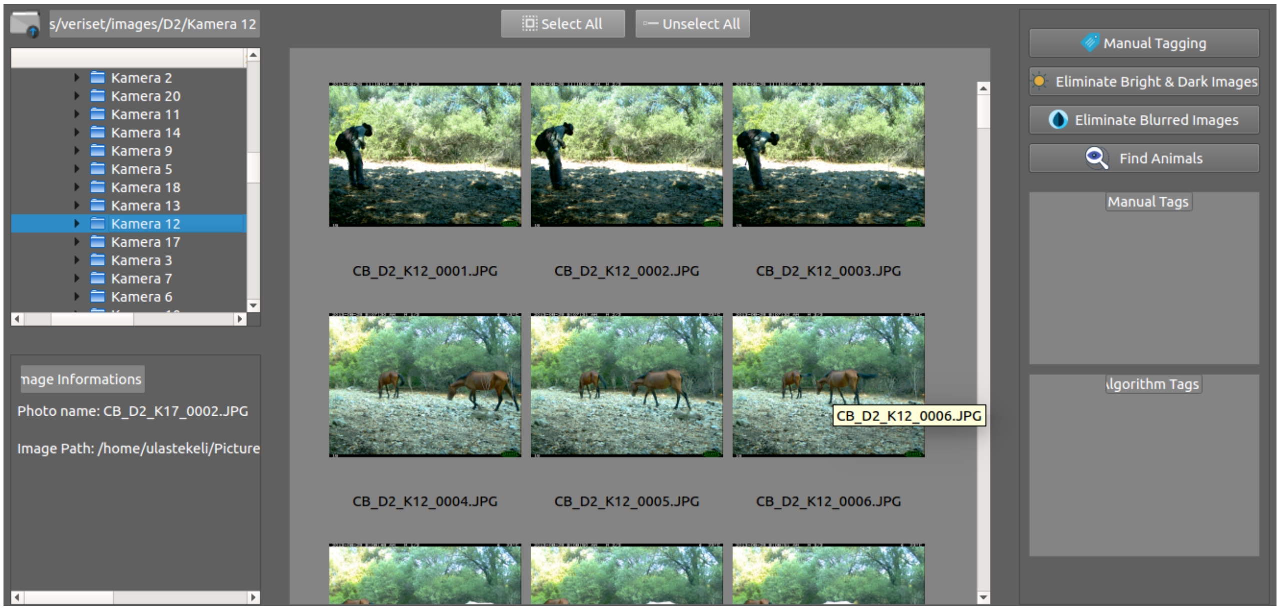


Figure 12: Screenshot from our prototype software. There is an explorer window on the left where user can select individual camera-traps. Images are shown in the middle panel where user can select one or multiple images. Right panel contains buttons for manual tagging and algorithm tagging (elimination methods). Also windows exist to view added tags.

7. Conclusions

Identifying animals in large sets of camera-trap images consumes a considerable amount of time for wild-life researchers. In Snapshot Serengeti project, annotating images collected in six months took more than two months by a group of 28,000 registered and 40,000 unregistered volunteers [30]. With the aim of reducing the number of camera-trap images to be visually examined by wildlife researchers, we developed different modules of image elimination. Blurred image elimination module worked with 94% accuracy with a cost of eliminating 11% of partially blurred photos. Too bright and too dark image elimination rate is 99% without eliminating any useful image.

Regarding animal/non-animal image classification, we employed an object detector CNN and kept images if any animal is found in images. Our approach reached an accuracy of 90.2%. We showed with experiments that this is well above the performance of state-of-the-art image classifier CNNs (which were used in previous work on camera-traps). Moreover our combined method achieved 99.1% remained image accuracy while obtaining 54.5% eliminated image accuracy. Overall accuracy seems to be low, but high remaining rate is preferred because penalty of a false-negative result is much higher (since that image will not be shown to the expert anymore).

Acknowledgment

This work was supported by the Scientific and Technological Research Council of Turkey (Grant no. 115E918). Authors are grateful to Republic of Turkey, Ministry of Forest and Water Affairs for sharing camera-trap dataset.

References

[1] Boom BJ, He J, Palazzo S, Huang PX, Beyan C, Chou HM, Lin FP, Spampinato C, Fisher RB. A research tool for long-term and continuous analysis of fish assemblage in coral-reefs using underwater camera footage. Ecological Informatics 2014; 23: 83-97.

[2] Song D, Xu Y. A Low False-Negative Filter for Detecting Rare Bird Species from Short Video Segments using a Probable Observation Data Set-based EKF Method. IEEE Transactions on Image Processing 2010; 19: 2321-2331.

[3] Weinstein BG. MotionMeerkat: Integrating motion video detection and ecological monitoring. Methods in Ecology and Evolution 2015; 6: 357-362.

[4] Hernández-Serna A, Jiménez-Segura LF. Automatic identification of species with neural networks. PeerJ. 2014; 2:e563. DOI:10.7717/peerj.563

[5] Yu X, Wang J, Kays R, Jansen PA, Wang T, Huang T. Automated identification of animal species in camera trap images. EURASIP Journal on Image and Video Processing 2013; 52.

[6] Chen G, Han TX, He Z, Kays R, Forrester T. Deep convolutional neural network based species recognition for wild animal monitoring. In: 2014 IEEE International Conference on Image Processing (ICIP), 27-30 October 2014, Paris, France; 858-862.

[7] Gomez-Villa A, Salazar A, Vargas F. Identification of animal species in camera-trap images using very deep convolutional neural networks. Ecological Informatics 2017; 41: 24-32.

[8] Norouzzadeh MS, Nguyen A, Kosmala M, Swanson A, Palmer MS, Packer C, Clune, J. Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. In: Proceedings of the National Academy of Sciences of the United States of America (PNAS) 2018; 115: E5716-E5725

[9] Nguyen H, Maclagan SJ, Nguyen TD, Nguyen T, Flemons P, Andrews K, Ritchie EG, Phung D. Animal Recognition and Identification with Deep Convolutional Neural Networks for Automated Wildlife Monitoring. In: IEEE International Conference on Data Science and Advanced Analytics (DSAA) 2017; 19-21 October 2017; Tokyo, Japan; pp. 40-49.

[10] Krishnappa YS, Turner WC. Software for minimalistic data management in large camera trap studies. Ecological Informatics 2014; 24: 11-16.

[11] Fegraus EH, Lin K, Ahumada JA, Baru C, Chandara S, Youn C. Data acquisition and management software for camera trap data: A case study from the TEAM network. Ecological Informatics 2011; 6: 345-353.

[12] Niedballa J, Sollmann R, Courtiol A, Wilting A. camtrapR: An R package for efficient camera trap data management. Methods in Ecology and Evolution, 2016, 7(12):1457–1462.

[13] Pavlovic G, Tekalp AM. Maximum likelihood parametric blur identification based on a continuous spatial domain model. IEEE Transactions on Image Processing 1992; 1(4): 496-504.

[14] Narvekar ND, Karam LJ. A no-reference image blur metric based on cumulative probability of blur detection (CPBD). IEEE Transactions on Image Processing 2011; 20(9): 2678-2683.

[15] Tong H, Li M, Zhang H, Zhang C. Blur detection for digital images using wavelet transform. In: 2004 IEEE International Conference on Multimedia and Expo (ICME), 27-30 June 2004, Taipei, Taiwan; 1: 17-20

[16] Dosselmann RW, Yang XD. No-Reference Noise and Blur Detection via the Fourier Transform, Technical Report, University of Regina, Canada, 2012.

[17] Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. In: International Conference on Neural Information Processing Systems 2012; 3-8 December 2012; Lake Tahoe, Nevada, USA; pp. 1097-1105.

[18] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC. ImageNet large scale visual recognition challenge. International Journal of Computer Vision 2015; 115(3): 211-252.

[19] Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, LeCun Y. Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv preprint 2013. arXiv:1312.6229.

[20] Ren S, He K, Girshick R, Sun, J. Faster R-CNN: Towards realtime object detection with region proposal networks. In: Advances in Neural Information Processing Systems (NIPS) 2015; 7-12 December 2015; Montreal, Canada; pp. 91-99.

[21] Redmon J, Divvala S, Girshick R, Farhadi A. You Only Look Once: Unified, Real-Time Object Detection. In: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) 2016; 27-30 June 2016; Las Vegas, Nevada, USA; pp. 779-788.

[22] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC. SSD: Single Shot MultiBox Detector. In: European Conference on Computer Vision (ECCV) 2016; 8-16 October 2016; Amsterdam, Netherlands; pp. 21-37.

[23] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016; 27-30 June 2016; Las Vegas, Nevada, USA; pp. 770-778.

[24] Orhan S, Bastanlar Y. Training CNNs with image patches for object localisation. Electronics Letters, 2018, 54(7): 424–426. DOI:10.1049/el.2017.4725

[25] Sobral A, Vacavant A. A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. Computer Vision and Image Understanding 2014, 122: 4-21.

[26] Zivkovic Z. Improved adaptive Gaussian mixture model for background subtraction. In: Proceedings of the 17th International Conference on Pattern Recognition ICPR 2004; 23-26 August 2004; Cambridge, UK; 2: 28-31

[27] Tabak MA, Norouzzadeh MS, Wolfson DW, et al. Machine learning to classify animal species in camera trap images: Applications in ecology. Methods in Ecology and Evolution. 2018; 1–6.

[28] Ju C, Bibaut A, van der Laan MJ. The relative performance of ensemble methods with deep convolutional neural networks for image classification. arXiv preprint 2017. arXiv:1704.01664v1.

[29] Islam J, Zhang Y. An ensemble of deep convolutional neural networks for Alzheimer's disease detection and classification. In: Machine Learning for Health Workshop at Neural Information Processing Systems (NIPS) 2017; 8 December 2017; Long Beach, CA, USA.

[30] Swanson A, Kosmala M, Lintott C, Simpson R, Smith A, Packer C. Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna. Scientific Data, 2015. DOI:10.1038/sdata.2015.26