ELIMINATION OF USELESS IMAGES FROM RAW CAMERA-TRAP DATA

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Computer Engineering

by
Ulaş TEKELİ

December 2018
İZMİR

We approve the thesis of **Ulaş TEKELİ**

Examining Committee Members:

Asst. Prof. Dr. Kaya OĞUZ
Department of Computer Engineering, İzmir University of Economics

Asst. Prof. Dr. Mustafa ÖZUYSAL
Department of Computer Engineering, İzmir Institute of Technology

Assoc. Prof. Dr. Yalın BAŞTANLAR
Department of Computer Engineering, İzmir Institute of Technology

26 December 2018

Assoc. Prof. Dr. Yalın BAŞTANLAR
Department of Computer Engineering
İzmir Institute of Technology

Assoc. Prof. Dr. Tolga AYAV
Head of the Department of
Computer Engineering

Prof. Dr. Aysun SOFUOĞLU
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

I would like to thank to my supervisor Assoc. Prof. Dr. Yalın Baştanlar for all his effort and patience he took with me in this learning process.

I would like to thank my friends from the Computer Vision Research Group for their support and give a special acknowlegdment to Semih Orhan for his support and insightful comments.

I also would like to thank Asst. Prof. Dr. Mustafa Özuysal and Asst. Prof. Dr. Kaya Oğuz, sparing their valuable time for evaluating this work.

I am grateful to my family for their love and support.

This thesis work is supported by The Scientific and Technical Research Council of Turkey (TUBITAK) with project number 115E918.

ABSTRACT

ELIMINATION OF USELESS IMAGES FROM RAW CAMERA-TRAP DATA

A common way to observe animals in nature is to use motion triggered cameras that are called camera-traps. With the expanding usage of camera-trap due to advances in digital technology, the number of images that are collected from camera-traps has increased significantly. Labeling and grouping of animals in these images have put enormous workload on wild-life researchers. We propose a system that frees time for researchers by eliminating useless images-too bright, too dark, too blurred images and images that contain no animals from raw camera-trap data. Firstly, we utilise image histograms to eliminate too bright and too dark images and Fast Fourier Transform to eliminate blurred ones. Secondly, we make use of deep learning techniques and background subtraction to eliminate images without animals and we present the result of our experiments on these subjects. Our approach on eliminating too bright and too dark images have missed very few images and on eliminating blur images we achieve 95.5% success. Finally we show that the technique we propose eliminates more than 50% of images without animals while containing 99% of images with animals.

ÖZET

HAM FOTOKAPAN VERİSİNDEN İŞE YARAMAZ İMGELERİN ELENMESİ

Doğadaki hayvanları gözlemlemenin en yaygın yollarından birisi fotokapan adlı hareket sensörlü kameralar kullanmaktır. Dijital teknoloji alanındaki gelişmelerle fotokapan kullanımının genişlemesi, fotokapanlardan toplanılan imge sayısında büyük bir artışa yol açmıştır. Bu imgeleri etiketleme ve gruplama görevleri, doğa araştırmacılarının üzerine büyük bir iş yükü bindirmiştir. Bu çalışmamızda aşırı karanlık, aşırı aydınlık, bulanık ve hayvan içermeyen imgeleri eleyerek araştırmacılar için zaman kazandıran bir sistem öneriyoruz. İlk olarak imge histogramlarından aşırı parlak ve aşırı karanlık imgeleri elemek için, Fast Fourier Transform'dan ise bulanık imgeleri elemek üzere faydalanıyoruz. İkincil olarak hayvan içermeyen imgeleri elemek için derin öğrenme ve arkaplan çıkarımı tabanlı bir yöntem kullanıyoruz ve bu konular üzerindeki deneylerimizin sonuçlarını sunuyoruz. Aşırı parlak ve aşırı karanlık imge eleme yaklaşımımız neredeyse hatasız çalışırken, bulanık imge eleme yaklaşımımız %95.5'lik bir başarı yakalamıştır. Son olarak kullandığımız tekniğin hayvan içermeyen imgelerin %50'sini elerken, hayvan içeren imgelerin %99'unu koruduğunu gösteriyoruz.

TABLE OF CONTENTS

LIST OF FIGURES

Figure **Page**

LIST OF TABLES

LIST OF ABBREVIATIONS

CNN . . . . . Convolutional Neural Network
ResNet . . . . . Residiual Neural Network
R-CNN . . . . . Regional Convolutional Neural Network
Fast R-CNN . . . . . Fast Regional Convolutional Neural Network
Faster R-CNN . . . . . Faster Regional Convolutional Neural Network
RPN . . . . . Region Proposal Network
CDF . . . . . Cumulative Distribution Function
ReLu . . . . . Rectified Linear Unit
ILSVRC . . . . . ImageNet Large Scale Visual Recognition Competition
SVM . . . . . Support Vector Machine
NMS . . . . . Non-Maximum Suppresion
YOLO . . . . . You Only Look Once
SSD . . . . . Single Shot MultiBox Detector

CHAPTER 1

INTRODUCTION

Camera-traps are motion-sensored cameras that are set up on the pathways of animals to observe animals in wild-life. Sample images that are captured from cameratraps are given in Fig.1.1. In recent years, cameratrap usage has been increasing continuously. A properly working camera-trap can capture nearly 1000 image in a month. Due to unexpected conditions (animals damaging the camera, heavy rainfall or storms) or cameras not working properly, a considerable amount of these images may be too dark, too bright or blur. Also the undistorted images may contain no animals. Since the aim of wild-life researchers is to examine these images and group or label them according to the species, above mentioned images can be considered useless. Images collected from high number of camera-traps takes too much time for researchers to examine and make use of.

Figure 1.1. Examples of images obtained from camera-traps.

1.1. Thesis' Aim and Objectives

The purpose of our study is to automatically exclude useless images from raw datasets of camera-traps and hence decrease the number of images to be visually checked. In our method, we firstly eliminate blurred, too bright and too dark images. To eliminate blurred images, we utilize a method based on Fast Fourier Transform and to eliminate too bright and too dark images, we employ image histograms. After the elimination of these images, the goal is to eliminate images without animals. (Fig. 1.2).

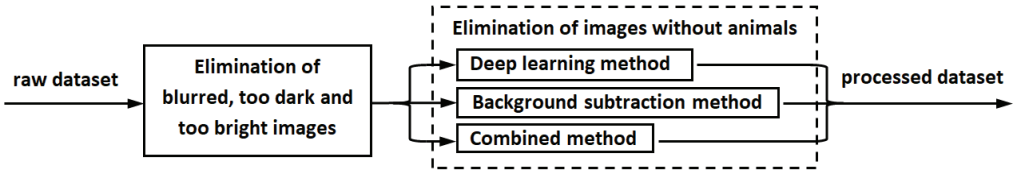


Figure 1.2. Pipeline of elimination process on raw camera-trap dataset.

For this purpose we evaluate two methods on animal detection in camera-trap images, first one is based on background subtraction (since the camera-traps accumulate images with varying time periods on the same background) and the second one uses convolutional neural networks (CNN). We further investigated how to combine these two approaches to obtain the best outcomes.

1.2. Organization of Thesis

In Chapter 2, we review the related work in literature and explain our contribution in detail. We present our methods of eliminating blurred, too bright and too dark images in Chapter 3. Chapter 4 is devoted to describe the methods of object detection in order to eliminate images without animals. We present experimental results in Chapter 5. Lastly, our conclusions are given in Chapter 6.

CHAPTER 2

RELATED WORKS

Studies on automatic animal detection and classification of images and videos that are gathered from nature are relatively new. In the work of Boom et al. (2014), a dataset was built with images collected from sub-aqua cameras and a large set of features including color, shape, texture properties and moment invariants are used. Fish classification was performed on the regions obtained by separating moving objects from the background. Other studies on distinguishing species includes the work of Dunn et al. (2003) where animals that are to display are photographed with a blue cloth in the background and the study of Hernández-Serna and Jiménez-Segura (2014) where the photographs from a museum database are used, therefore animals are easily distinguished from background.

Song and Xu (2010) conducted one of the studies to decrease the workload of wild-life researchers. In this work, birds are detected on a video and tracked with a Kalman Filter. Standard Kalman Filter is adapted in a way that considers the constraints related to birds speed or where the bird may be in an image for better tracking of birds. This way authors aim to show only the videos with a high probability of containing birds to the experts that will examine the video. With a similar purpose, Weinstein (2015) detects moving objects in videos captured in nature, and show the experts only the frames that contains any moving objects.

The first study to use convolutional neural networks (CNN) to classify species from camera-trap images was conducted by Chen et al. (2014). They use the dataset of the University of Missouri that contains 20 species. Although CNN carries much more potential on this task, they achieve a disappointing 38% accuracy because this study took place in 2014 (early years of CNN's). Villa et al. (2017) tested different CNN structures with a much bigger dataset (Snapshot Serengeti, 26 species, 780.000 images) reported an accuracy of 60%. Norouzzadeh et al. (2018) made another study on Snapshot Serengeti dataset, training the CNN models from scratch. This increased classification accuracy up to 94% with the best model when the consideration is the highest probability class (top-1 accuracy). In this study, two class (animal, non-animal) image classification is

also performed and 96.8% accuracy is achieved with the best performing model. Another study which used a different but again large camera-trap dataset was conducted by Nguyen et al. (2017) and reported 90.4% accuracy for species classification and 96.6% accuracy for animal/non-animal classification.

Our intent in this work is not to classify species, rather do animal/non-animal classification and our result for eliminating images without animals are comparable with animal/non-animal classification results in the literature. Prior studies that obtained up to 96% accuracy on this task (Norouzzadeh et al. (2018); Nguyen et al. (2017)) were held with very big and diverse sets of camera-trap images. Our dataset is more challenging in the way that we work with raw image folders (a folder per camera-trap) and we do not mix train and test folders (train and test images are from different locations/backgrounds) which satisfies the real-life scenario where test images come from new camera-trap locations. Under these conditions, according to our experiments, a state-of-the-art image classification CNN (ResNet) reached only 80.7% accuracy on animal/non-animal classification. We present our first contribution by training an object detector CNN (Faster R-CNN) to separate images that contains animals from the images that does not and eliminate the latter ones. Furthermore, we propose a process that combines CNN and background subtraction methods. In our experiments, we show that this combined method achieved 99.1% rate of keeping photos with animals, therefore minimizing the loss of important data, while eliminating more than 50% of images without any animal. To the best of our knowledge, our study is the first study that combines CNN and background subtraction to eliminate camera-trap images without animals.

Besides eliminating images without animals, as mentioned before, we also proposed algorithms to eliminate too dark, too bright and blurred images. For blur detection, we proposed an approach to discriminate partially blurred images from completely blurred ones. By partitioning the image to subimages, we decide if an image has a clear area that may contain animals. And for too bright and too dark detections, we employ an empirical threshold on the bright and dark scores obtained from image histograms.

CHAPTER 3

BLURRED, BRIGHT AND DARK IMAGE ELIMINATION

3.1. Blurred Image Elimination

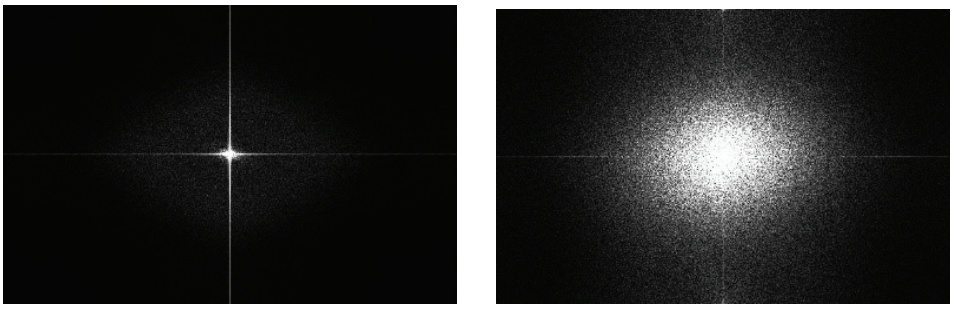
In the last 25 years, many approaches on blur detection in images have taken their places in the literature. Pavlovic and Tekalp (1992) propose an approach that utilizes maximum likelihood on spatial space to recognize blur. Narvekar and Karam (2011) employ a cumulative probability metric, whereas Tong et al. (2004) use wavelet transformation based on edge shapes and edge sharpness.

Another method that is commonly used on blur detection is Fourier Transform. The Fourier Transform is a tool that breaks a waveform (a function or signal) into an alternate representation, characterized by sine and cosines. In other words, it converts an image to frequency domain from the spatial domain. On the centered spectrum obtained by Fourier Transform, low-frequency coefficients are represented close to the center and the farther away from the center, the higher frequency coefficients are placed. Since the intensity differences between adjacent pixels of a blurred image are too low, a blurred image must produce a spectrum with very low frequencies (accumulation in the center). Fig.3.1 shows two images that are labeled as blurred and clear and their corresponding Fourier spectra.

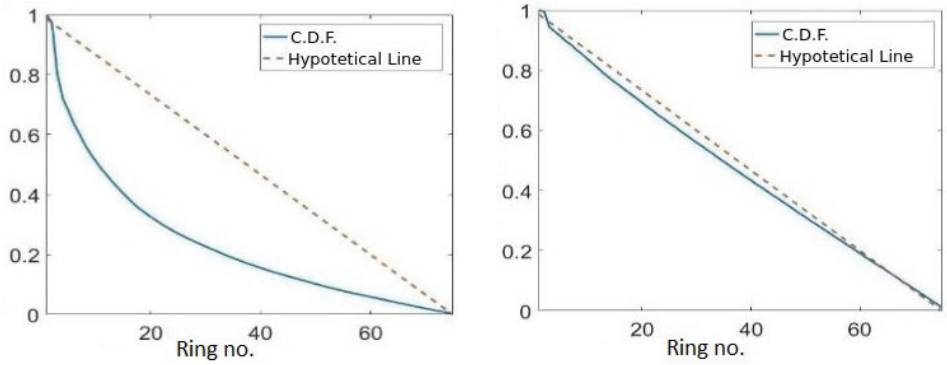
Dosselmann and Yang (2012) place rings with a varying radius on the Fourier spectrum's center and calculate the responsiveness of areas between rings. The sum of pixels values between each ring is recorded and used to form a cumulative distribution function (CDF) shown in Fig. 3.1 (last column). The 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 a CDF value of 1.0 when the ring number is 1. Also, a hypothetical line is shown in the figure, representing an image with an 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 a lower ϕ than a threshold are labeled as blurred.

(a)



(b)



(c)

Figure 3.1. (a) Left: A blurred image, Right: A clear image. (b) Fourier transforms of images from (a). (c) Cumulative distribution function (CDF) of the images from (a) with the algorithm given in Dosselmann and Yang (2012).

The algorithm described above (Dosselmann and Yang, 2012) 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. Examples of partially blurred images can be seen in Fig.3.2. We propose an

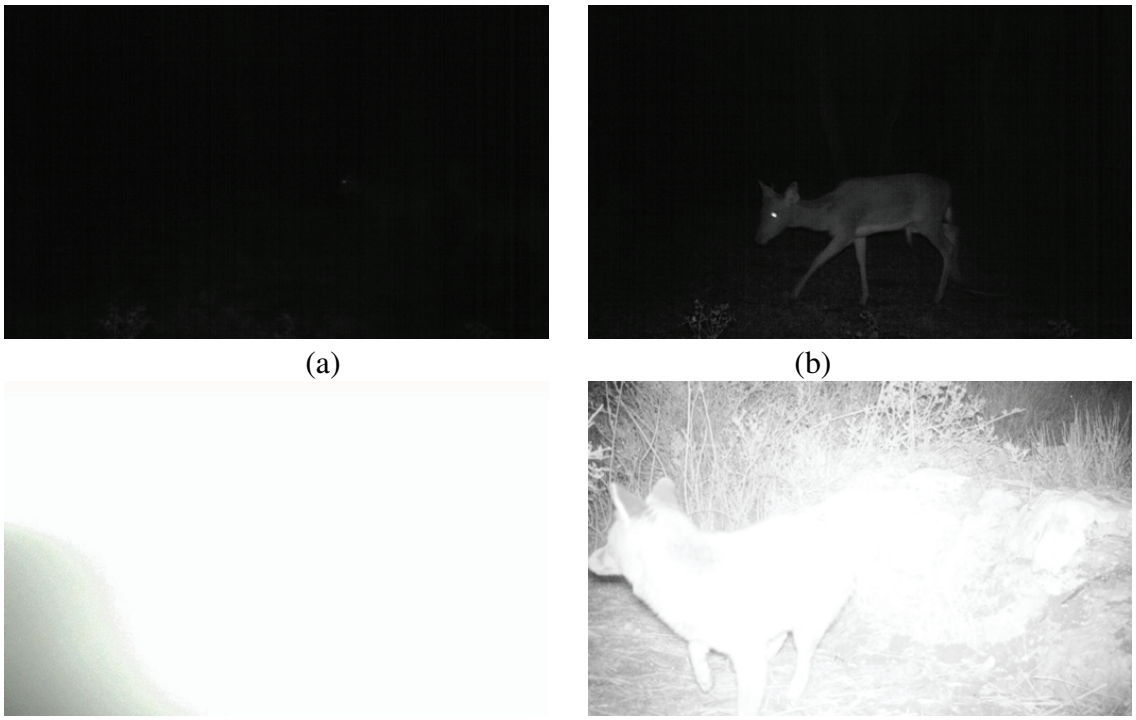
approach based on the work of Dosselmann and Yang and compute blurriness in different parts of images. If only a few parts of the image are blurred, it is not eliminated. We divide the images into a fixed number of sub-images and for each sub-image we perform blur detection. The number of blurred sub-images 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 as a result of decreased image sizes 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 Fig.3.3. Equation 3.1 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 $\leq$20 are dark and pixels with intensity value $\geq$220 are bright, where intensity range is [0,255]. These are empirical values based on our observations on the dataset. These observations are obtained by exhaustive searching of the spatial domain space. We tested the proposed formula from 128 to 255 for bright images and from 0 to 128 for dark images and choose the best performing values as a final decision. Also, thresholds on p_d and p_b were set as 0.94 and 0.84 respectively. Again these empirical threshold values are outcomes of an exhaustive search. We checked the target space from 0.5 to 1 increasing 0.01 at a time for both bright and dark image testing and come up with these thresholds. Images, whose darkness or brightness is higher than these thresholds, are eliminated.

$$p_d = \left(\frac{\sum_{i=0}^{20} hist(i)}{\sum_{i=0}^{255} hist(i)}\right)^2 \qquad p_b = \left(\frac{\sum_{i=220}^{255} hist(i)}{\sum_{i=0}^{255} hist(i)}\right)^2 \tag{3.1}$$

Figure 3.2. Partially blurred images in raw camera-trap dataset



(a) (b)

(c) (d)

Figure 3.3. Examples of too dark (a), dark but useful (b), too bright (c) and bright but useful (d) images.

CHAPTER 4

DETECTING IMAGES WITH ANIMALS

4.1. Deep Learning

Convolutional Neural Network is a type of artificial neural network that is widely used for image recognition tasks. It assumes there is a high correlation between image pixels and using convolutions on multiple pixel values instead of using each pixels seperately as in fully connected layers, would take advantage of this correlations. Because of this feature, CNNs are also good at feature extraction out of correlated data inputs.

A CNN consists of input, output and hidden layers. The hidden layers of CNN (Fig. 4.1) generally consists of 4 layers: Convolutional layer and Rectified Linear Unit (ReLU), pooling layer, normalization layer and fully connected layers.

In convolution layers, a convolution operation with some kernel (for example 3x3 convolution matrix) is applied to input in a sliding window fashion and the result is passed to the next layer. Each neuron in convolutional layer processes only its receptive field data in contrast to neurons in fully connected layers. To put it simply, convolution maps a region of an image to a feature map. This way the number of learnable parameters decreases significantly and vanishing or exploding gradient problems that are encountered in a traditional multi-layer neural networks are prevented. ReLU is an element-wise operation to replace negative values in the input map with zeros. The purpose of ReLU is to introduce non-linearity to CNN's since convolution is a linear operation and the real world data usually is non-linear. ReLU is applied after every convolution layer.

The pooling layer (Fig. 4.2), is a sub-sampling strategy that is used to reduce the spatial dimensions in order to decrease the number of parameters and amount of computation, and hence to also control overfitting. Max pooling and average pooling are the most common pooling strategies. Pooling divides the input map into a set of rectangles and outputs the maximum or average value for every rectangle, creating an output map smaller than the input map. For example the most common pooling layer filter is of size 2x2, which shrinks the input map to its quarter size.

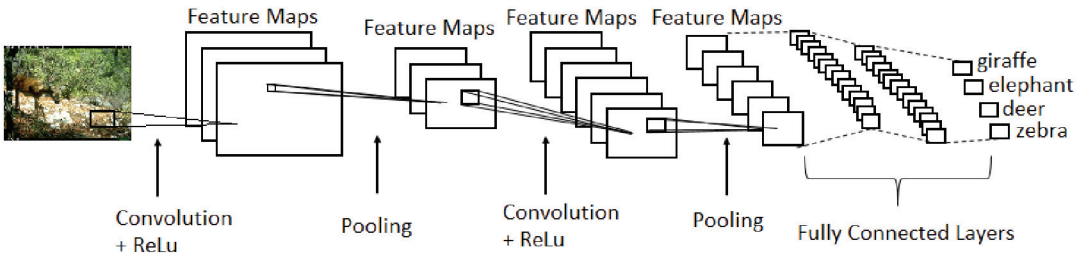


Figure 4.1. A generic CNN structure. (Source: Prabhu, Medium.Com, 2018)

The output from the convolutional and pooling layers represent high-level features of the input image. The purpose of the Fully Connected layer is to use these features for classifying the input image into various classes based on the training dataset. Apart from classification, adding a fully-connected layer is also a (usually) cheap way of learning non-linear combinations of these features. Most of the features from convolutional and pooling layers may be good for the classification task, but combinations of those features might be even better.

Convolutional Neural Networks (CNN), especially after AlexNet (Krizhevsky et al., 2012) won the ILSVRC (Russakovsky et al., 2015) competition of image classification in 2012, have been effectively used on many tasks of computer vision, including object detection as well as image classification. There are quite a few CNN approaches developed for object detection. Since we propose to use an object detector CNN, let us quickly review some of those.

OverFeat (Sermanet et al., 2014), an earlier object detection approach, trains a CNN with both, classifier and regressor heads, and search objects in images with a sliding window. R-CNN (Girshick et al., 2014) uses selective search to generate 2000 region proposals. Then the generated proposals are extracted from the image and warped into square and finally fed into CNN seperately. Feature map produced by CNN is used to classify by a Support Vector Machine (SVM). In addition to the classifying objects, R-CNN also has a regressor head that does bounding box regressions that produces 4 offset values to make the proposed area more precise. One problem with this approach is that it still took a huge amount of time to train or test 2000 proposals for an image. The other problem was the fixed region proposal algorithm. Since there is no learning on generating the region proposal, the classifier CNN depends on selective search to produce accurate predictions.

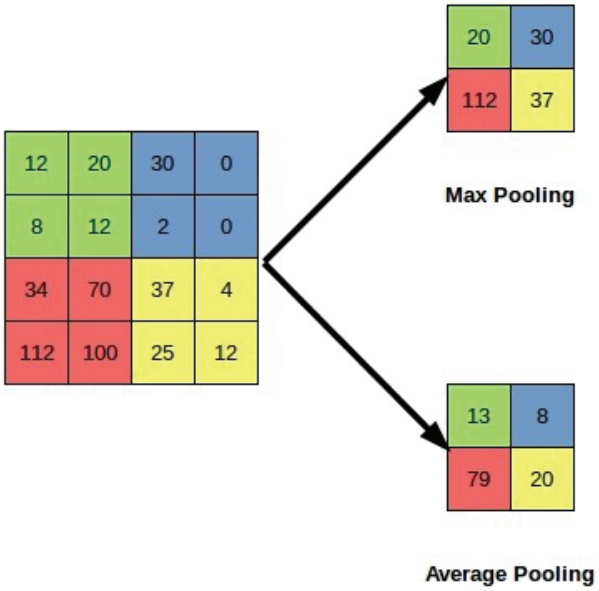


Figure 4.2. Pooling Example.

To overcome the first problem, authors proposed a slightly different approach with Fast R-CNN (Girshick, 2015). Proposals still are produced with selective search but instead of feeding all proposals to CNN seperately, all of the image fed into CNN at once and a convolutional feature map is generated. From the convolutional feature map, the region of proposals are identified and warped into squares. Then, by using a region of interest (RoI) pooling layer they are reshaped into a fixed size so that they can be fed into a fully connected layer.(Fig. 4.4) Although this approach speeds up significantly from R-CNN, selective search still took 2 seconds per image and it was still a fixed algorithm that must be depended on. Hence the authors proposed yet another approach: Faster R-CNN.

Faster R-CNN (Ren et al., 2015) employs a Region Proposal Network (RPN) which shares last convolutional layer of proven CNN (VGG-16 in this case) with a classifier network attached to it to make object proposals (Fig. 4.5). This last shared convolutional layer acts as a feature map for both networks. RPN produces nearly 20000 region proposals with different sizes and scales and assigns an objectness score to each proposal. Then RPN uses non-maximum suppression (NMS) to decrease the number of proposals by eliminating intersecting ones and to further eliminate low score proposals, it applies a score threshold and ends up with top-N (300 is chosen in the article (Ren et al.,

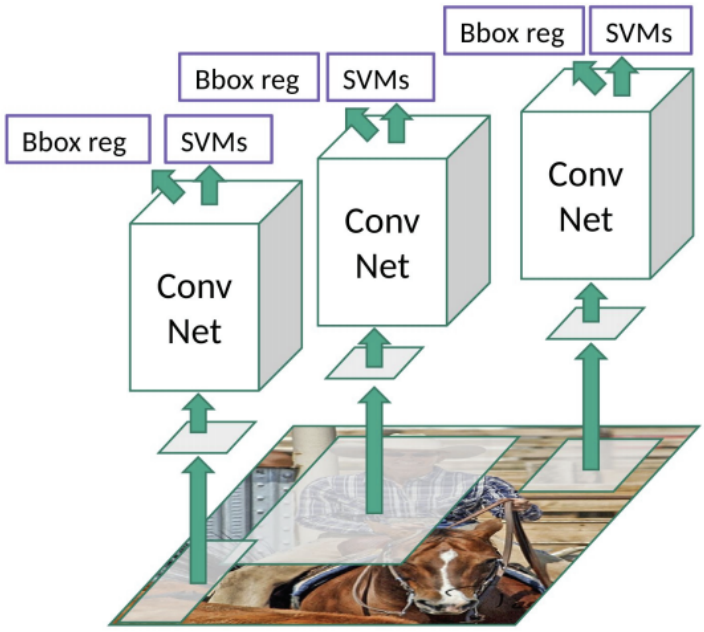


Figure 4.3. R-CNN structure. (Source: Girshick et al., 2014)

2015)) proposals. CNN is fed with remaining top-N proposals and it performs classification on the receptive fields of these regions located in the last shared convolutional layer and bounding box regression on the proposal windows. By making proposals with a neural network instead of traditional computer vision techniques, Faster R-CNN has gained significant speed improvements.

YOLO (Redmon et al., 2016) and SSD (Liu et al., 2016) takes another road 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 at once. Although with the significant speed improvement over Faster R-CNN, early attempts on YOLO did not manage to reach Faster R-CNN's accuracy while SSD has also replicate Faster R-CNN success on accuracy. Recent improvements on YOLO also reached the detection accuracy of Faster R-CNN while processing real-time.

We chose Faster R-CNN for our object detection module. Its proven effectiveness on different datasets and the abundance of documentation and source codes are the main reasons of this choice.

We trained Faster R-CNN as a two class classifier since our aim is detecting images with animals and not classifying species and we fed the network with a training data where all images consists of images with animals. This is due to the architecture of object

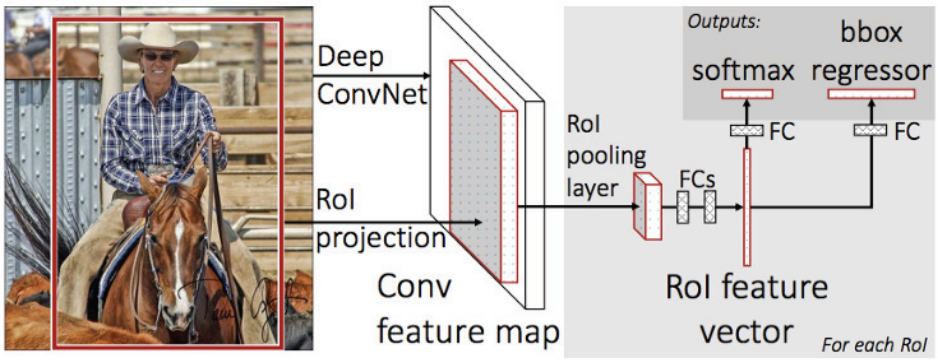


Figure 4.4. Fast R-CNN structure. (Source: Girshick, 2015)

detection networks as they classify regions and an image with an animal also contains much more background regions. During training we make use of transfer learning. First we use pretrained weights on ImageNet dataset for the backbone architecture which is VGG16. Then we initialize weights of RPN and classifier fully connected layers with a standard deviation of 0.01 and mean of 0 and start training. We observe that the model trained with the whole network from scratch does not converge as well as the model that uses transfer learning.

We must also clarify why we chose an object detector instead of simply using a classifier network for animal/non-animal image classification. The reason is that general purpose image classifiers such as ResNet (He et al., 2016) does not perform well enough for our dataset obtained from Ministry of Forest and Water Affairs. Our interpretation of this observation is that since image classifiers looks at the whole image to make a judgement, it must be confusing to seperate two image with the same background, where in one of them there is an animal that took small part of image and in other there is no animal. 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 (Chen et al., 2014; Villa et al., 2017; Norouzzadeh et al., 2018), 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 4.1 shows the comparison of performance between state-of-

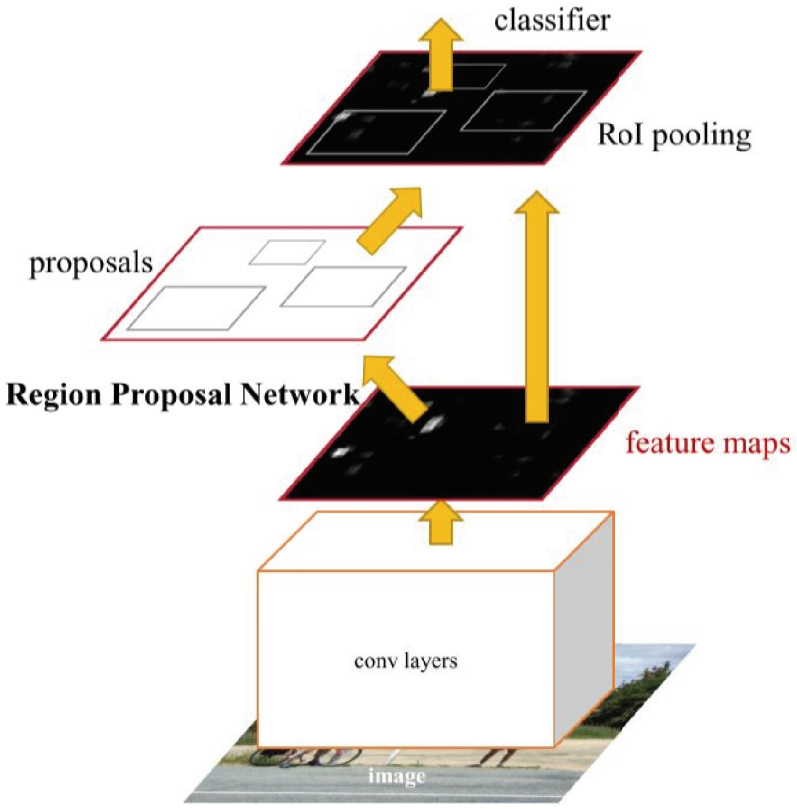


Figure 4.5. Faster R-CNN structure. (Source: Ren et al., 2015)

the-art classification network ResNet and Faster R-CNN 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 4.1. ResNet (He et al., 2016) and Faster R-CNN (Ren et al., 2015) 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. Background Subtraction

Background subtraction is a well-known approach to detect moving objects in real-time videos. Since our camera-trap image sequences exhibit similarity to videos in

terms of low change rate between frames, we decided to evaluate this approach. Camera-traps collect images with varying time intervals on the same scene, resulting in a long image sequence with a single background.

Sobral and Vacavant (2014) presents a comprehensive review of background subtraction algorithms in his survey. We preferred to use the Gaussian Mixture Model (Zivkovic, 2004) 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

$$p(x) = \sum_{j=1}^{K} w_j \mathcal{N}(x, \theta_j) \tag{4.1}$$

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.

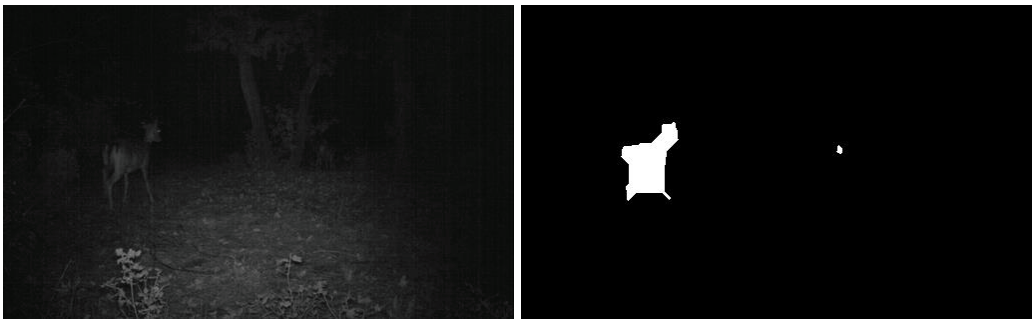
Static single-color objects tend to form tight clusters in the color space while moving ones form wider clusters due to different reflecting surfaces during the movement. Thus, w_k/σ_k is used as the fitness value to represent staying long and more tight. Higher fitness value refers to having higher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

$$B = \underset{b}{\operatorname{argmin}}(\sum_{j=1}^{b} w_j > T) \tag{4.2}$$

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 Zivkovic (2004), 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.

Fig.4.6a shows a successful example of a component defined as object.

Failures usually happen when lighting rapidly changes between two consecutive images (an example is given in Fig.4.6b). Since camera-trap image sequence is collected



(a)

(b)

Figure 4.6. 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.

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.

First, we construct 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 rate of the changed pixels establishes our similarity metric. A low rate indicates a 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 group sorted images from where

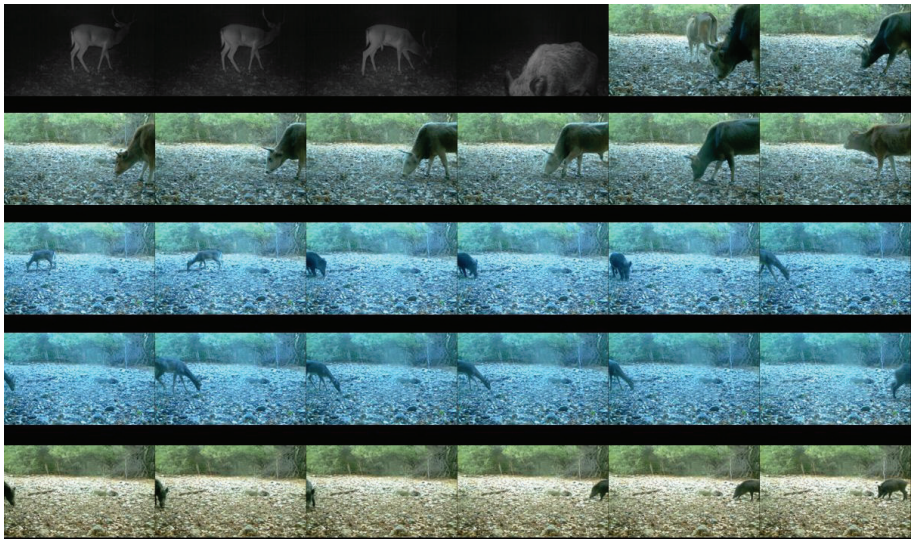


Figure 4.7. Clusters that show up after sorting the images. Starting from top-left, 1st, 5th, 13th, and 25th images are starting points of new clusters.

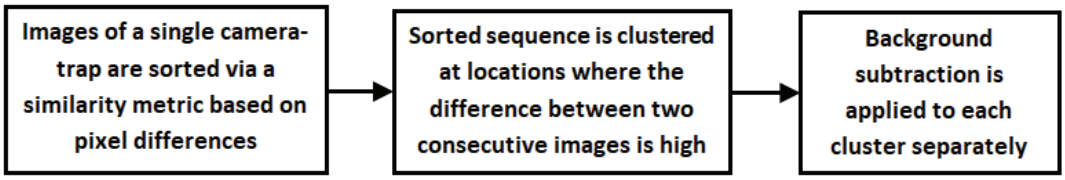


Figure 4.8. Steps of the proposed pre-processing for background subtraction approach.

the lighting changes significantly (low similarity between consecutive images) on image series. We see 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 Fig.4.7. Later, we apply background subtraction to each group 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 Fig.4.8.

The proposed sorting algorithm improves the results since it decreases the number of the failures, especially the ones similar to Fig.4.6b. 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.

CHAPTER 5

EXPERIMENTAL RESULTS

We obtain our raw camera-trap dataset that consists of nearly 40000 camera-trap images from the Ministry of Forest and Water Affairs, Republic of Turkey. These images are collected from different cameras at different times and stored in a way that each folder in raw dataset comes from a distinct background scene. The images from the same scene differs in capturing time up to few months. First we manually search through the images to label blurred, too bright and too dark images. Then we created bounding-box annotations to 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 5.1 shows the accuracy of the classification of images following the approach explained in Section 3.1. Out of 186 blurred images, 175 are labeled as blurred, achieving 94.1% accuracy. 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. 94.1% blur elimination is good since it saves human time, but it comes with a cost of eliminating 11% of partially blurred photos.

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 5.2. Only 11 dark images are incorrectly classified, no errors made on bright and useful images. Thus, this module

Table 5.1. Blurred image detection results

Actual Classes	# of images	Proposed Approach			Original Approach		
		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%

Table 5.2. 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

is more effective than the blur elimination since it eliminates 99% of useless photos with no false-negatives.

5.2. Experiments on Eliminating Images without Animals

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 5.3. 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 5.3). 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 (Chen et al., 2014). 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.

Table 5.3. 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

5.2.1. Deep Learning

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. This approach is also a good choice for the situations where one can encounter animals which do not exist in the training set.

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 5.3). Results are shown in Fig.5.1 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 5.4 shows the detailed result of the experiment when threshold is kept at 0.5. On the average of two datasets, average of 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 5.5. Both eliminated and remained image

Table 5.4. 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%

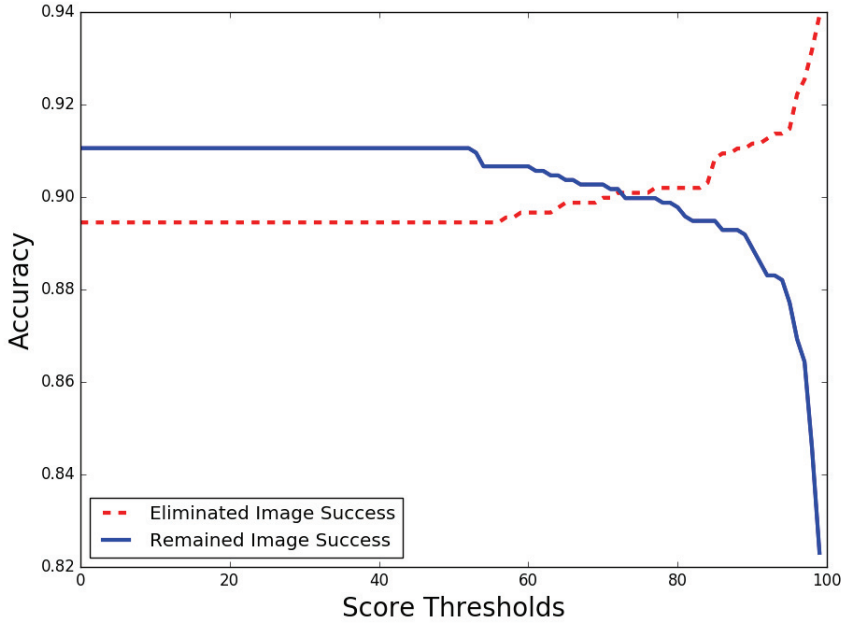


Figure 5.1. Results on deep learning experiments with different score thresholds.

accuracies show a decline, pointing out that the generalization capacity of a model trained with a camera-trap dataset from a single source is limited.

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 (Ju et al., 2018; Islam and Zhang, 2017). For this purpose, we trained four separate networks to be used as the classifier of Faster R-CNN model (Fig. 4.5) 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.

Test set consisting of both DS-1 and DS-2. Results are shown in Table 5.6 where baseline learner refers to the single Faster R-CNN that uses 100% of training data. When

Table 5.5. 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 5.6. 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%

we compare baseline learner and ensemble of networks, we observe a small improvement in total accuracy as expected.

5.2.2. 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. Fig.5.2 shows some example detections. Some animals were missed due to the similarity of their texture with the background (Fig. 5.2c), whereas some large stones are mistaken as animals (Fig. 5.2d). 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 subtraction to each cluster of images on its own. The experiment results on DS-1 and DS-2 are given in Table 5.7. Our first observation is that the eliminated rate drops on both sets when compared to deep learning (cf. Table 5.4). This indicates that background subtraction causes a higher number of false-positives. On the other hand, for DS-2, remained rate increased to 75% (cf. Table 5.4) catching most of the animals that are missed by deep learning method.

Table 5.7. 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%

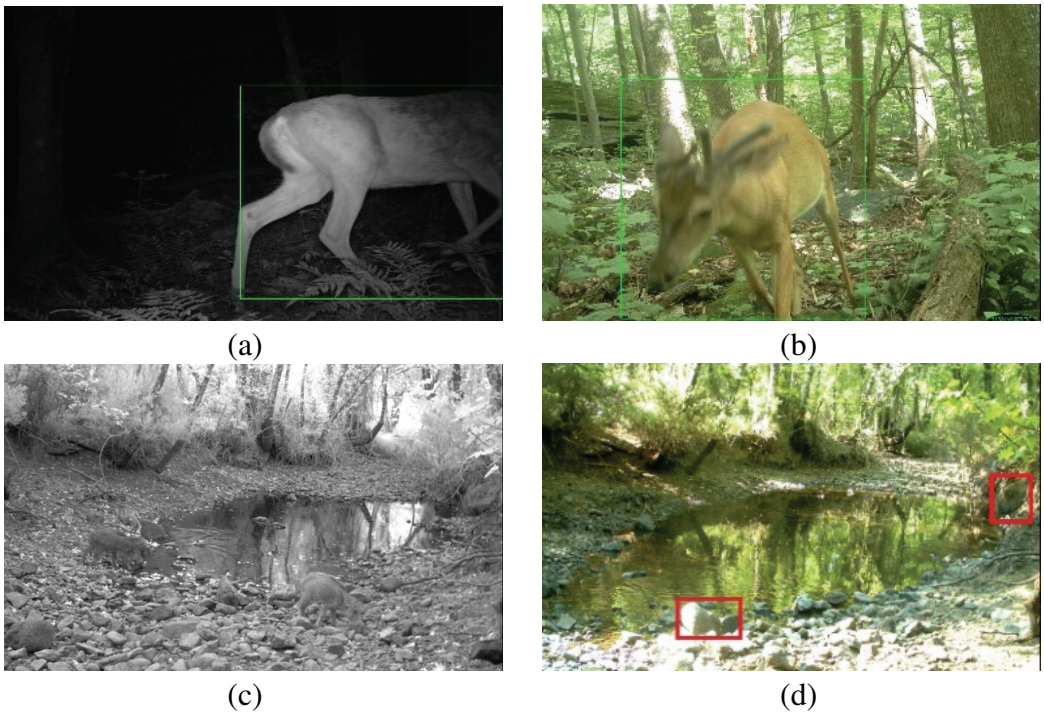


Figure 5.2. Some examples of correct detections (a,b), missed animals (c) and false-positive detections (d) with deep learning method.

5.2.3. Combined Method

Although background subtraction method's results are not as successful as deep learning's, we observed that the two methods generally fail on different images. Therefore, we decided to perform an experiment that combines the decisions of both methods. 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 5.8 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.

Table 5.8. 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%

CHAPTER 6

CONCLUSIONS

We aimed to decrease the workload of wild-life researchers by reducing the number of camera-trap images to be visually examined. 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).

REFERENCES

Boom, B., J. He, S. Palazzo, P. X. Huang, C. Beyan, H.-M. Chou, F.-P. Lin, C. Spampinato, and R. B. Fisher (2014). A research tool for long-term and continuous analysis of fish assemblage in coral-reefs using underwater camera footage. *Ecological Informatics 23*, 83–97.

Chen, G., T. X. Han, Z. He, R. Kays, and T. Forrester (2014). Deep convolutional neural network based species recognition for wild animal monitoring. In *2014 IEEE International Conference on Image Processing (ICIP)*, pp. 858–862.

Dosselmann, R. and X. D. Yang (2012). No-reference noise and blur detection via the fourier transform. Technical report, Department of Computer Science, University of Regina, Regina, SK, CANADA.

Dunn, M., J. Billingsley, and N. Finch (2003). Machine vision classification of animals. In *10th Annual Conference on Mechatronics and Machine Vision in Practice*.

Girshick, R. (2015). Fast r-cnn. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, Washington, DC, USA, pp. 1440–1448. IEEE Computer Society.

Girshick, R., J. Donahue, T. Darrell, and J. Malik (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, Washington, DC, USA, pp. 580–587. IEEE Computer Society.

He, K., X. Zhang, S. Ren, and J. Sun (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Hernández-Serna, A. and L. F. Jiménez-Segura (2014). Automatic identification of species with neural networks. *PeerJ 2*, e563.

Islam, J. and Y. Zhang (2017). An ensemble of deep convolutional neural networks for alzheimer's disease detection and classification. *In Machine Learning for Health Workshop at Conference on Neural Information Processing Systems (NIPS 2017) abs/1712.01675*.

Ju, C., A. Bibaut, and M. van der Laan (2018). The relative performance of ensemble methods with deep convolutional neural networks for image classification. *Journal of Applied Statistics 45*(15), 2800–2818.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, USA, pp. 1097–1105. Curran Associates Inc.

Liu, W., D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg (2016). Ssd: Single shot multibox detector. In B. Leibe, J. Matas, N. Sebe, and M. Welling (Eds.), *Computer Vision – ECCV 2016*, Cham, pp. 21–37. Springer International Publishing.

Narvekar, N. D. and L. J. Karam (2011). A no-reference image blur metric based on the cumulative probability of blur detection (cpbd). *IEEE Transactions on Image Processing 20*(9), 2678–2683.

Nguyen, H., S. J. Maclagan, T. D. Nguyen, T. Nguyen, P. Flemons, K. Andrews, E. G. Ritchie, and D. Phung (2017). Animal recognition and identification with deep convolutional neural networks for automated wildlife monitoring. In *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 40–49.

Norouzzadeh, M. S., A. Nguyen, M. Kosmala, A. Swanson, M. S. Palmer, C. Packer, and J. Clune (2018). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences 115*(25), E5716–E5725.

Pavlovic, G. and A. M. Tekalp (1992). Maximum likelihood parametric blur

identification based on a continuous spatial domain model. *IEEE Transactions on Image Processing 1*(4), 496–504.

Prabhu, Medium.Com (2018). Understanding of convolutional neural network (cnn) – deep learning. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148. [Online; accessed 14-November-2018].

Redmon, J., S. K. Divvala, R. B. Girshick, and A. Farhadi (2016). You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788.

Ren, S., K. He, R. Girshick, and J. Sun (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, Cambridge, MA, USA, pp. 91–99. MIT Press.

Russakovsky, O., J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision 115*(3), 211–252.

Sermanet, P., D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun (2014). Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR2014), CBLS, April 2014*.

Sobral, A. and A. Vacavant (2014). A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos. *Computer Vision and Image Understanding 122*, 4 – 21.

Song, D. and Y. Xu (2010). 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 19*(9), 2321–2331.

Tong, H., M. Li, H. Zhang, and C. Zhang (2004). Blur detection for digital images using wavelet transform. In *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, Volume 1, pp. 17–20 Vol.1.

Villa, A. G., A. Salazar, and F. Vargas (2017). Towards automatic wild animal monitoring: Identification of animal species in camera-trap images using very deep convolutional neural networks. *Ecological Informatics 41*, 24 – 32.

Weinstein, B. G. (2015). Motionmeerkat: integrating motion video detection and ecological monitoring. *Methods in Ecology and Evolution 6*(3), 357–362.

Zivkovic, Z. (2004). Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, Volume 2, pp. 28–31 Vol.2.