INTELLIGENT ALERT SYSTEM FOR FOREST TRIBAL

using Convolutional Neural Network

Developed by: katta abhishta,suram anjana,salendra sangeetha,pitta mukesh

Smart Bridge-Remote Summer Internship Program

I.INTRODUCTION

Deep learning is a subset of machine learning. Deep artificial neural networks are a set of algorithms that have set new records in accuracy for many important problems, such as image recognition, sound recognition, etc., In deep learning, a convolutional neural network (CNN) is a class of deep neural networks, most commonly applied to analyzing visual imagery.CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage. They have applications in image and video recognition, recommender systems, image classification, medical image analysis, and natural language processing. One of the applications of the deep learning technique called Convolutional Neural Network is animal detection. Observing wild animals in their natural environments is a central task in ecology. The fast growth of human population and the endless pursuit of economic development are making over-exploitation of natural resources, causing rapid, novel and substantial changes to Earth's ecosystems. An increasing area of land surface has been transformed by human action, altering wildlife population, habitat and behaviour. More seriously, many wild species on Earth have been driven to extinction, and many species are introduced into new areas where they can disrupt both natural and human systems.
Monitoring wild animals, therefore, is essential as it provides researchers evidences to inform conservation and management decisions to maintain diverse, balanced and sustainable ecosystems in the face of those changes.

1.1OVER VIEW

When any animal comes within the range of this camera, the system should detect and identify the animal. This is a challenging task because, since the camera is always

monitoring the forest the image taken using this may or may not contain the presence of animal. So our system should be capable of filtering out the images which contain animal, the central and challenging task of the system is that it should recognize the species of the animal in which it is belongs. Existing technologies for wildlife monitoring include Radio tracking satellite tracking, Global Positioning System (GPS) tracking, Very High Frequency (VHF) and sensor networks. As technology advances many machine learning approachesare used for this purpose. Convolutional neural networks, Deep convolutional neural networks are the main approaches available for image classification and recognition. But to use CNN or Deep CNN as image classifier we require very large dataset. Even thousands or ten thousands of labelled images in each class may not be enough to train a convolutional neural network from the scratch. It is not always practical to get such a huge dataset and training the neural network with such a huge set of images is tedious and it take days or even months to train the system. So training a convolutional neural network from the scratch is not practical to our project because our system should be capable of completing the training with in hours and the animal should be detected as soon it comes in the range of camera. To meet this requirement traditional CNN is not appropriate. So a new concept in machine learning called Transfer learning is used here. i.e. in this paper animal classification and recognition is achieved by performing transfer learning on a pre-trained neural network by using collected labelled images. When the system detects the presence of animal it produces an alarm to inform the people and the forest rangers. Furthermore it sends a alert message which contain the name of detected of animal to the people in the locality and to the authority of forest. But while considering the security of people this is not enough because animal crossing the border can create lot of problem such as destruction of agricultural fields, killing of domestic animals, destruction of vehicle trespassing through the area and even cause for the loss of human life. So here further a system is designed which helps to repel animal back to the forest. Ultrasonic sensor is used for this purpose. We know that an ultrasonic sensor continuously produce ultrasonic wave. The frequency of ultrasonic wave is about 40 kHz, which is beyond the

audible range of human being and animal can easily hear this sound. Which create a hostile and noisy environment for the animal and by hearing this noisy sound animal get repel back to the forest. In this paper, the section II describes a detailed literature survey followed by Result and conclusion has been made

1.2 PURPOSE

The purpose of this is to explain the alerting them and Tribals especially with special reference to Tribal people. Tribal communities live in forests, and therefore, in India they are called 'janjatis'
(forest dwellers). Their socio-cultural life is mostly woven around nature.
Forest trees and common property resources are basic to tribal communities, directly benefit them like a foster mother and fulfill their biological, cultural, religious and emotional needs. For food, tribals are mostly dependent on forest by collecting nuts, wild fruits, vegetables, leaves, flowers, roots, stems, honey, wild animal and insects etc. Therefore, for their security it is used

2. LITERATURE SURVEY

In this paper, a novel system for automatic detection and classification of animal is presented. System called ASFAR (Automatic System For Animal Recognition) is based on distributed so-called 'watching device' in designated area, the main task of watching device is to detect animals in wild nature and then send the descriptions to main computing unit (MCU) for evaluation. it act as server and system. Video camera, computation unit, control unit, communication unit and supply unit are the main parts of watching device. manager.

The tasks of MCU are:

• Collecting images from watching devices
• Using classification algorithms to assess unknown objects to defined classes
• Determine migration corridors,

• store all results,

• control and manage watching devices and other eqipments.

Watching device collect data from designated area then, the important task of automatic system is to create motion vectors of animals and the corresponding migration corridors. ASFAR will be placed in wild nature, often without access to the electricity network and internet connection. System needs to work 24 hour a days and as long as possible. Therefore, there is a need to minimize the power consumption of the system. It is necessary to systematize a detection algorithm and effective object description algorithms to reduce data transfer over communication module. One of the main tasks of MCU is evaluate unknown object received from watching device to known classes. To perform this task, there is need to use methods for object recognition and classification. This method consists of two parts, training and testing part. In training part, visual descriptors are extracted from training image dataset and they are used to create a classification model. In testing part, classification model is used to evaluate an unknown objects to the appropriate class. Visual descriptors are used to capture the local appearance of objects. In ASFAR system, combination Bags of visual key points (BOW) and Support Vector Machine (SVM) methods were used to create a classification model. First, training data collections are used to set up the classification model parameters to distinguish different classes. Then, the classifier is able to evaluate an unknown object to the appropriate class.

2.1. EXISTING PROBLEM

Agriculture is the most important sector of Indian Economy but the issue of damage to crops by wild creatures has turned into a noteworthy social issue in current occasions. So far there is no effective solution to this problem and therefore requires earnest consideration. . This project provides a smart solution to resolve this problem. In this framework, image is captured when an animal intrudes and then image is classified as domestic or wild animal using Convolution Neural Network (CNN) and deep learning technique. This classification helps in alerting the farmer by alerting in case of intrusion of wild animal. The smart farm protection system gives reliable security and safety to crops. This system guarantees the wellbeing of creatures while warding them off. It

likewise diminishes the exertion made by man in securing the field and All the tribal populations of India were traditionally closely associated with forests, and there are some who even today spend the greater part of their lives in the proximity of trees and villages or clans near to forest. If any dangerous predators when entered into a village or clan may lead to loss of resources or in extreme cases leads to loss of life. Here we come up with an artificial intelligence based technique which detects the animals before its gets enter into villages. If any wild animals entered or detected this system identifies the animals and alerts the people. This ensures complete safety of humans who lives near the forests.

2.2 PROPOSED SOLUTION

The whole system can be divided in to three main part

1. Animal classification and recognition from real time video.

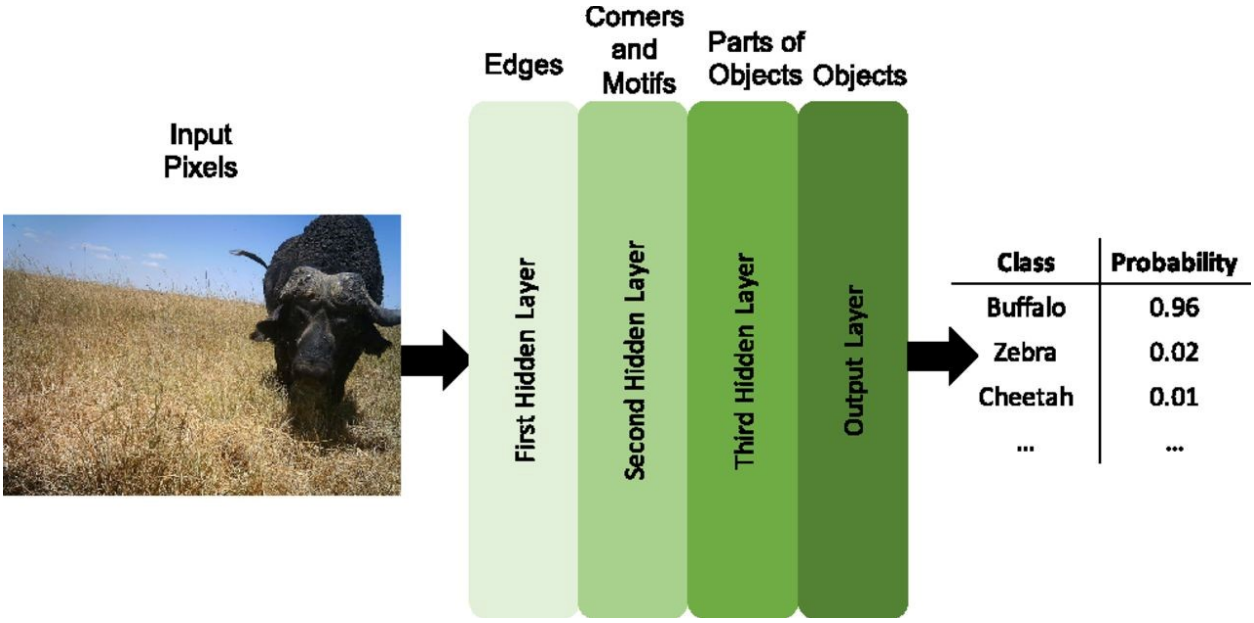
2. Alarm unit

A. Real time Animal classification and recognition

en the literature large amount of approaches have been proposed to accomplish the task of animal recognition that has different aims, strengths and limitations. Here new approach called ”Transfer learning” is used for animal detection and recognition. Before going deep in to transfer learning need to know about neural network and Convolutional Neural Network.

1) Convolutional neural network: A simple CNN is a sequence of layers. It mainly uses three main types of layers to build CNN architectures. That are Convolutional Layer, Pooling Layer, and Fully-Connected Layer. Stacking of these layers gives a full Convolutional neural network architecture. a) Convolutional Layer: The Convolution layer is the core building block of CNN that does most of the computations. The Convolution layer consist of a set of learnable filters. Every filter is small spatially (along width and height), but extends through the full depth of the input volume. For example, a typical filter on a first layer of a Convolution layer might have size 3 × 3 × 3 (i.e. 3 pixels width and height, and 3 colour channels). We convolve by sliding each filter across the width and height of the input volume and compute dot products between the filter coefficient

and the input at any position. As we slide the filter over the input volume we get a 2-dimensional activation map, that gives the responses of that filter at every position. ie, the network will learn feature such as an edge of some orientation or a blotch of some colour on the first layer, or eventually entire texture or body part on higher layers of the

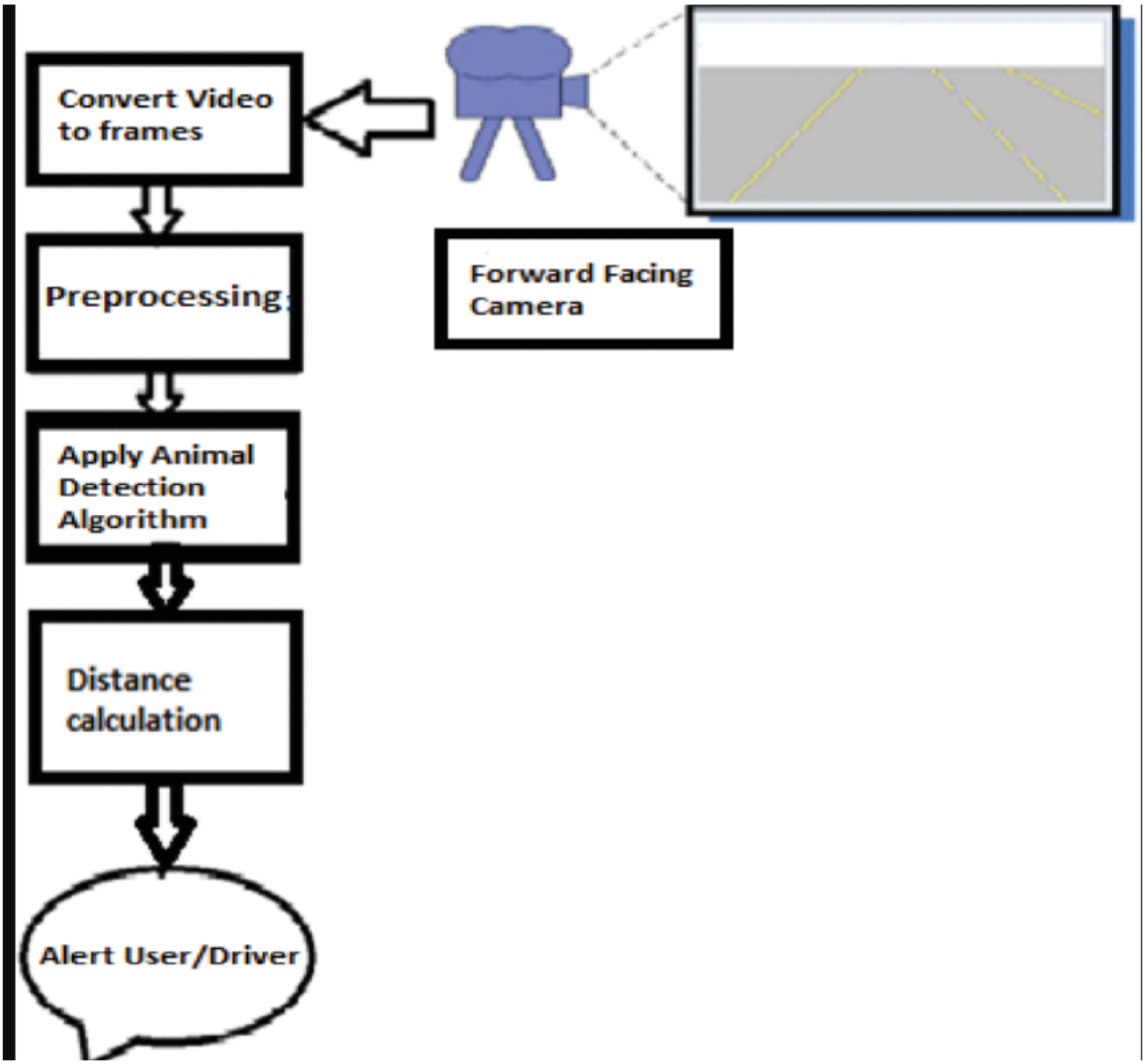


network. Intuitively, will have number of filters in Convolution layer and each of them will produce a separate two dimensional activation map. At the end we take up all the activation map and put them together as the output of convolution layer. When dealing with images, it is impractical to connect neurons to all neurons in the previous volume. Instead, we will connect each neuron to only a local region of the input volume. b) Pooling Layer: Its function is to progressively reduce the spatial dimension of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2×2 applied with a stride of two,

every depth slice in the input by 2 along both width and height, discarding 75% of the activations. c) Fully-connected layer: Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset. CNN is composed of two major parts Feature Extraction and Classification Fig.1 shows the architecture of CNN. The pink circle inside the red dotted region named classification is the neural network or multi-layer perceptron which acts as a classifier. The inputs to this network is given from the preceding part named feature extraction. Feature Extraction: This is the part of CNN architecture from where this network derives its name. Convolution is the mathematical operation which is central to the efficacy of this algorithm. The input to the red region is the image which we want to classify and the output is a set of features. Think of features in an image, for example, an image of a tiger might have features like whiskers, two ears, four legs etc. Convolution in CNN is performed on an input image using a filter or a kernel. The filter slides over the input image one pixel at a time starting from the top left. The filter multiplies its own values with the overlapping values of the image while sliding over it and adds all of them up to output a single value for each overlap. Similarly we compute the other values of the output matrix. Note that the top left value, which is 4, in the output matrix depends only on the 9 values (3 × 3) on the top left of the original image matrix. It does not change even if the rest of the values in the image change. This is the receptive field of this output value or neuron in our CNN. Each value in our output matrix is sensitive to only a particular region in our original

ge. After sliding our filter over the original image the output which we get is passed through another mathematical function which is called an activation function. The activation function usually used in most cases in CNN feature extraction is ReLU which stands for Rectified Linear Unit. Which simply converts all of the negative values to 0 and keeps the positive values the same. Once you get the feature maps, it is common to add a pooling or a sub-sampling layer in CNN layers. Pooling reduces the dimensionality to reduce the number of parameters and computation in the network. This shortens the

training time and controls over-fitting. The most frequent type of pooling is max pooling, which takes the maximum value in a specified window. The windows are similar to our earlier kernel sliding operation. This decreases the feature map size while at the same time keeping the significant information. Classification: All classification tasks depend upon labeled datasets; that is, humans must transfer their knowledge to the dataset in order for a neural network to learn the correlation between labels and data. 2) Transfer learning: The ability to use a pre-trained model as a shortcut to learn patterns from data it was not originally trained on. The beauty of deep learning lies in the fact, that pretrained model can be used to classify entirely different sets of data. This internally uses the pretrained weights of these deep neural net architectures (trained on ImageNet dataset) to apply on our own dataset. We use a network which is pretrained on the imagenet dataset and this network has already learnt to recognize the trivial shapes and small parts of different objects in its initial layers. By using a pretrained network to do transfer learning, we are simply adding a few dense layers at the end of the pretrained network and learning what combination of these already learnt features help in recognizing the objects in our new dataset. Hence we are training only a few dense layers. Furthermore, we are using a combination of these already learnt trivial features to recognize new objects. All this helps in making the training process very fast and require very less training data compared to training a convolution network from scratch. B. Alarm unit and Mobile application When the system detect the presence of any animal in the border region, the system produce an alarm and also it send an alert msg which contain the name of detected animal to the people living in the border region. Which helps human beings to take early precaution from the intrusion of wild animal.

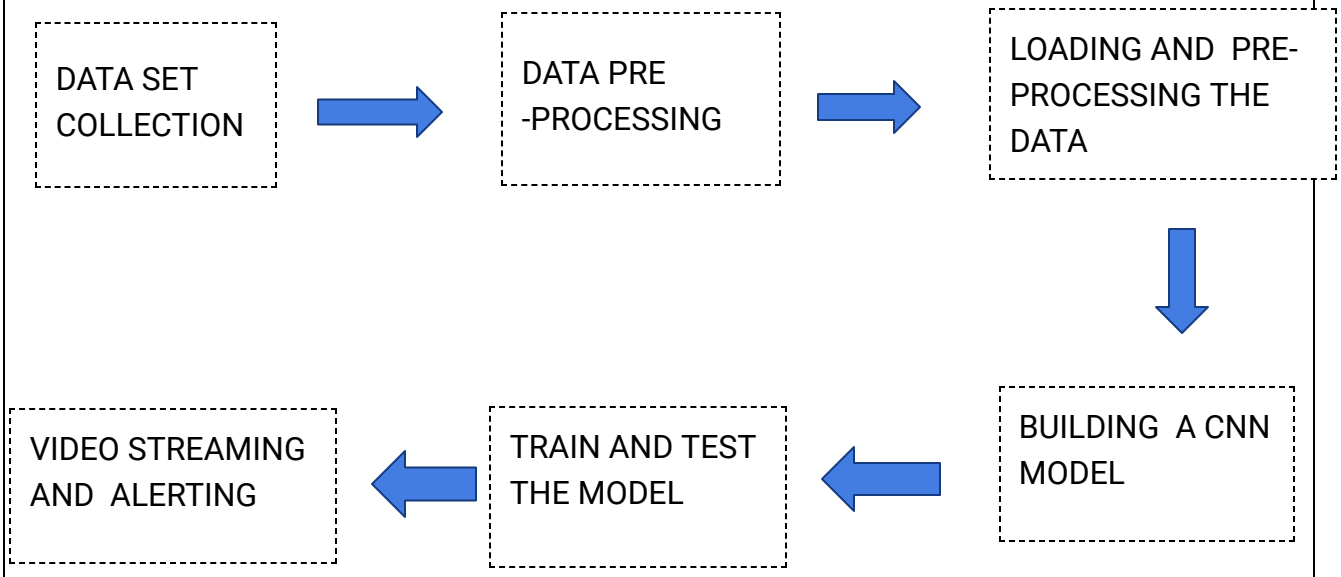


.

3.THEORITICAL ANALYSIS

Reliable and robust wildlife detection from highly dynamic and cluttered image sequences of camera-trap network is a challenging task. Hence to gain high performance, images need to be analyzed at pixel or small region level. However, due to low contrast and cluttered images, it becomes difficult to identify whether a particular region or pixel based on local information represents animal or background. Hence, we need to analyze global image features also. For example, the region of an animal body may be counted as background region. In such case, local information processing will not be sufficient, leading to requirement of global processing (to extract global image features) to detect animal. For example, recognize whether an animal is present or not, one should also identify body parts like head, legs etc. rather than body only.

3.1 BLOCK DIAGRAM

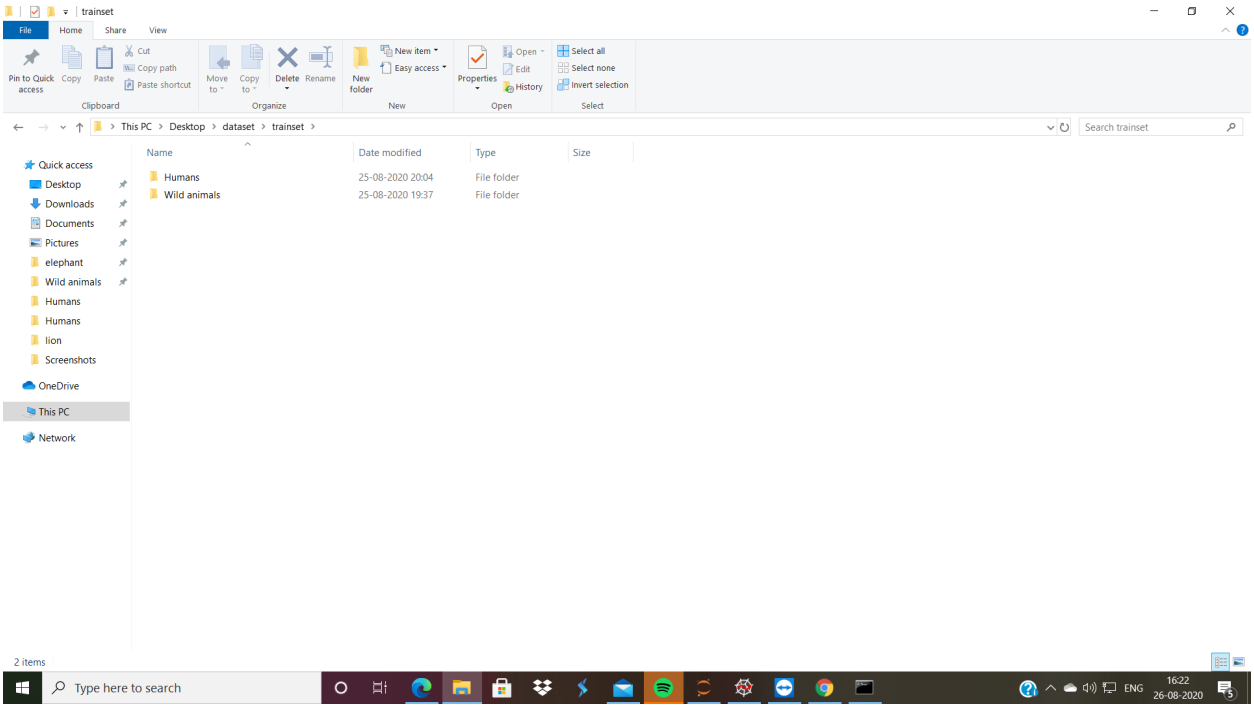


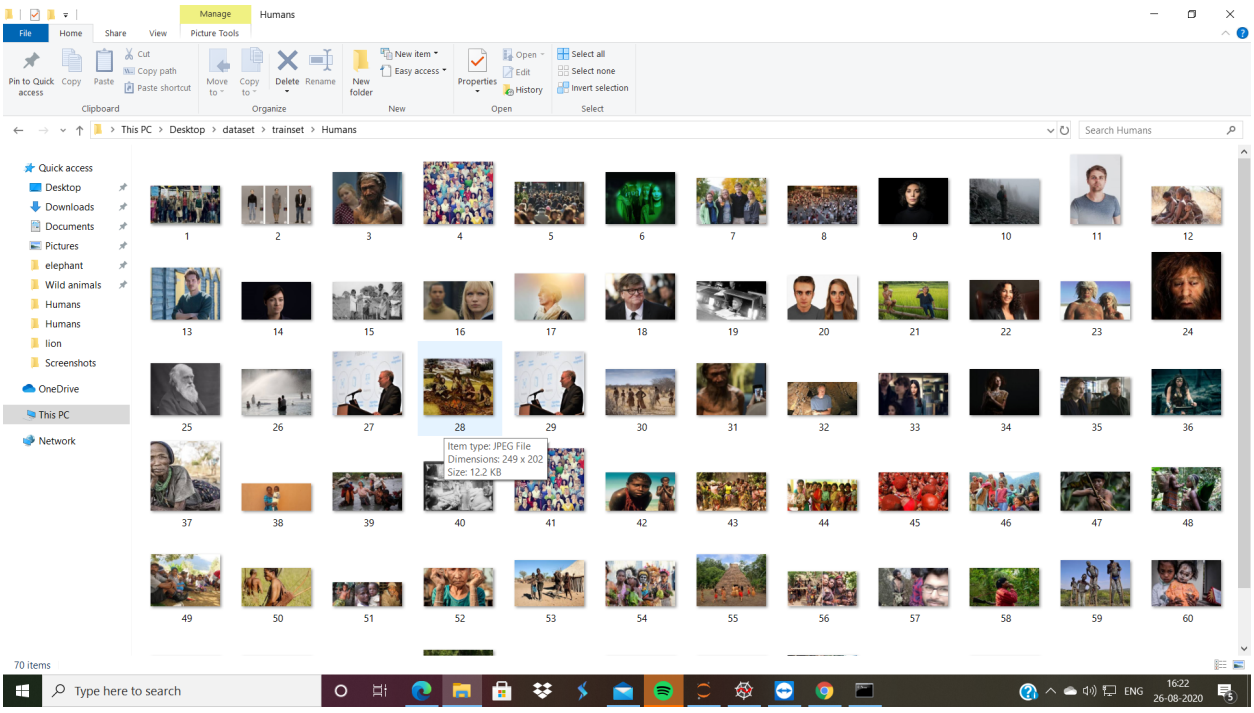
3.2 SOFTWARE DESIGNING

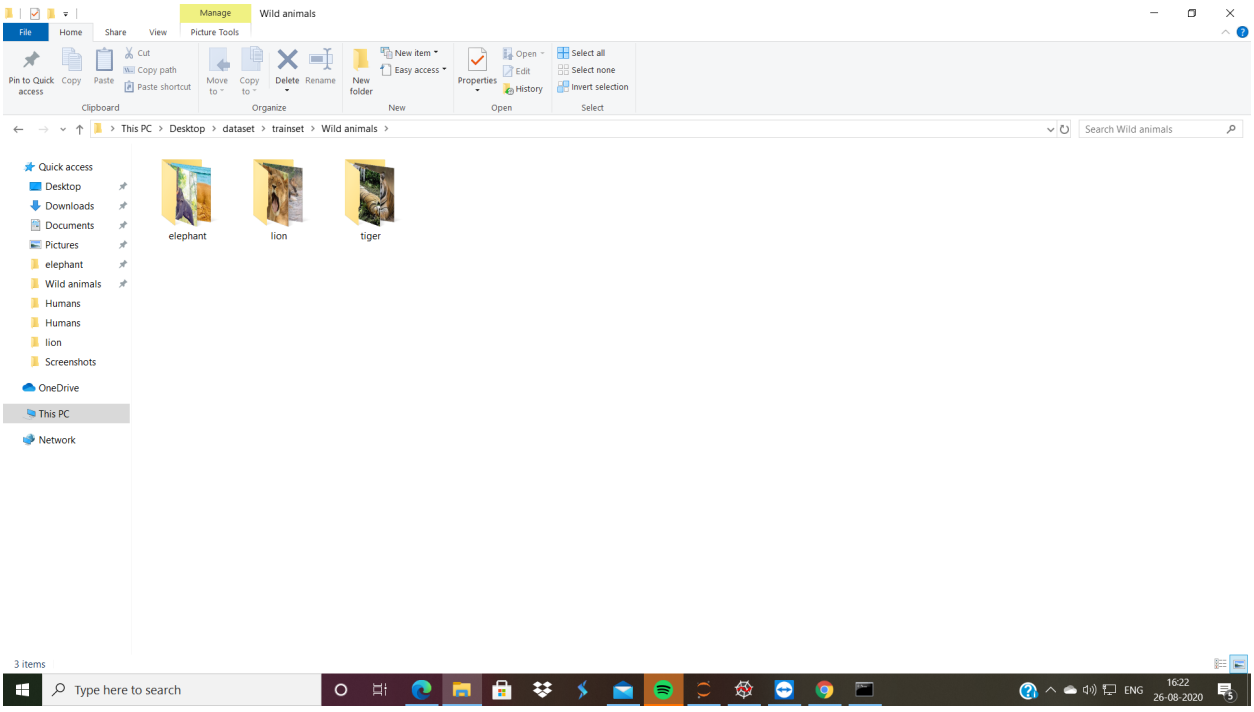
- Jupyter Notebook Environment
- Spyder Ide
- Machine Learning Algorithms
- Python (pandas, numpy, matplotlib, seaborn, sklearn)

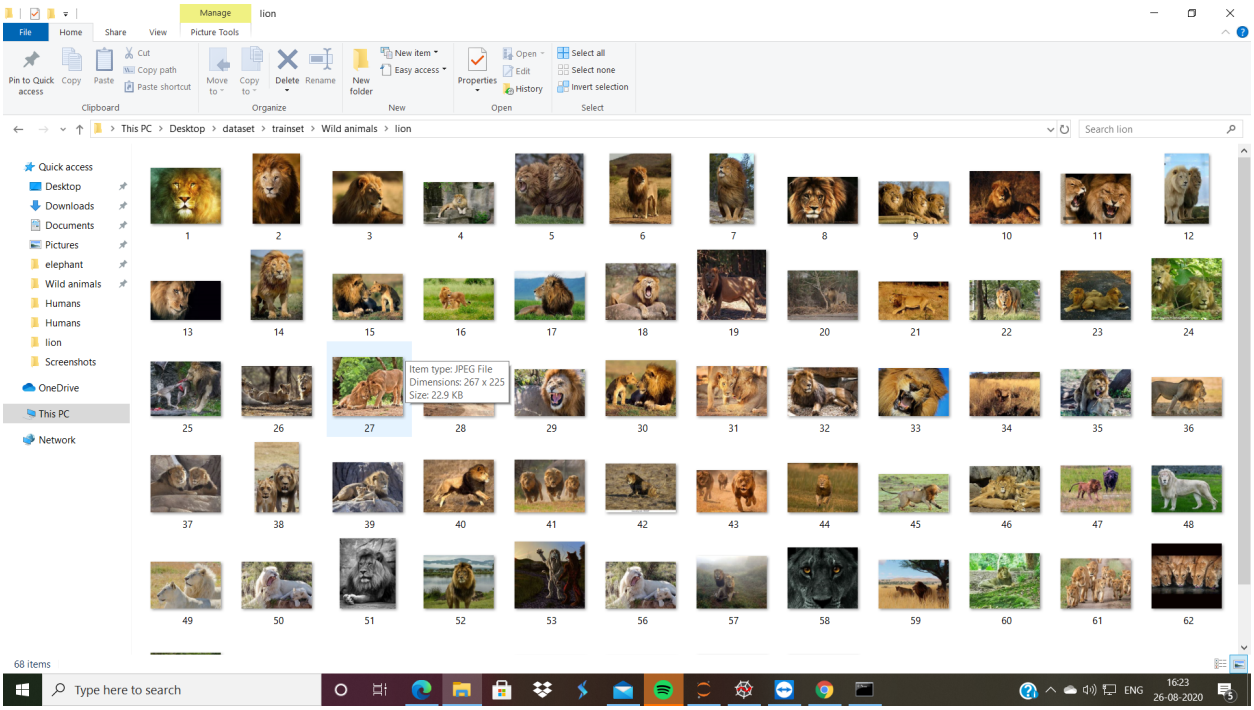
- HTML
- Flask

4. EXPERIMENTAL INVESTIGATION

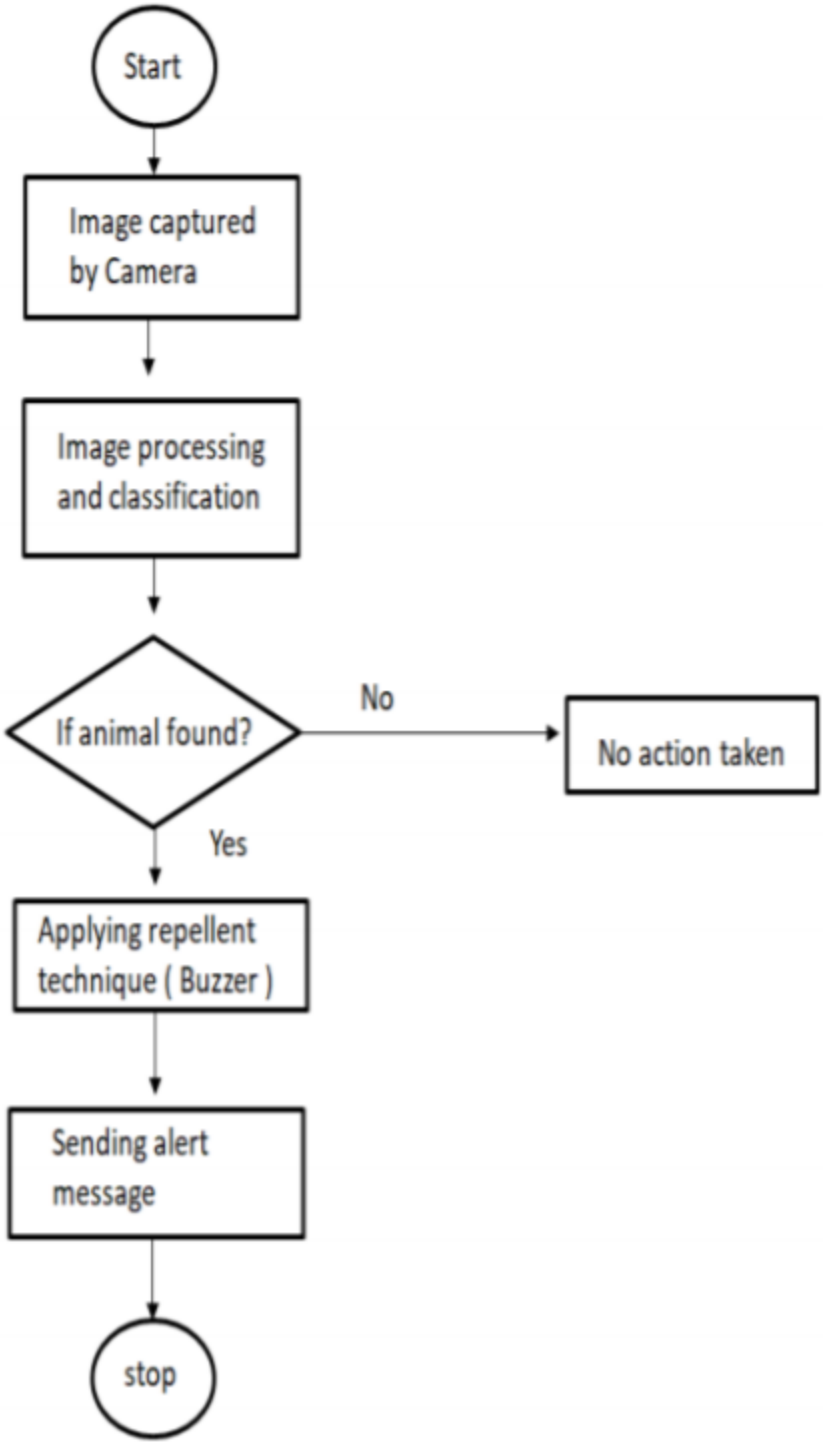



Manage
Humans
File
Home
Share
View
Picture Tools
Pin to Quick access
Copy
Paste
Cut
Copy path
Paste shortcut
Clipboard
Move to
Copy to
Delete
Rename
Organize
New folder
New item
Easy access
New
Properties
Open
Edit
History
Open
Select all
Select none
Invert selection
Select
This PC > Desktop > dataset > trainset > Humans
Search Humans
Quick access
Desktop
Downloads
Documents
Pictures
elephant
Wild animals
Humans
Humans
lion
Screenshots
OneDrive
This PC
Network
Item type: JPEG File
Dimensions: 249 x 202
Size: 12.2 KB
70 items
Type here to search
ENG
16:22
26-08-2020


Manage
Wild animals
File
Home
Share
View
Picture Tools
Pin to Quick access
Copy
Paste
Cut
Copy path
Paste shortcut
Clipboard
Move to
Copy to
Delete
Rename
Organize
New folder
New item
Easy access
New
Properties
Open
Edit
History
Open
Select all
Select none
Invert selection
Select
This PC > Desktop > dataset > trainset > Wild animals >
Search Wild animals
Quick access
Desktop
Downloads
Documents
Pictures
elephant
Wild animals
Humans
Humans
lion
Screenshots
OneDrive
This PC
Network
elephant
lion
tiger
3 items
Type here to search
ENG
16:22
26-08-2020


lion
Manage
File
Home
Share
View
Picture Tools
This PC
Desktop
dataset
trainset
Wild animals
lion
Search lion
Quick access
Desktop
Downloads
Documents
Pictures
elephant
Wild animals
Humans
Humans
lion
Screenshots
OneDrive
This PC
Network
Item type: JPEG File
Dimensions: 267 x 225
Size: 22.9 KB
68 items
Type here to search
ENG
16:23
26-08-2020

5.FLOWCHART

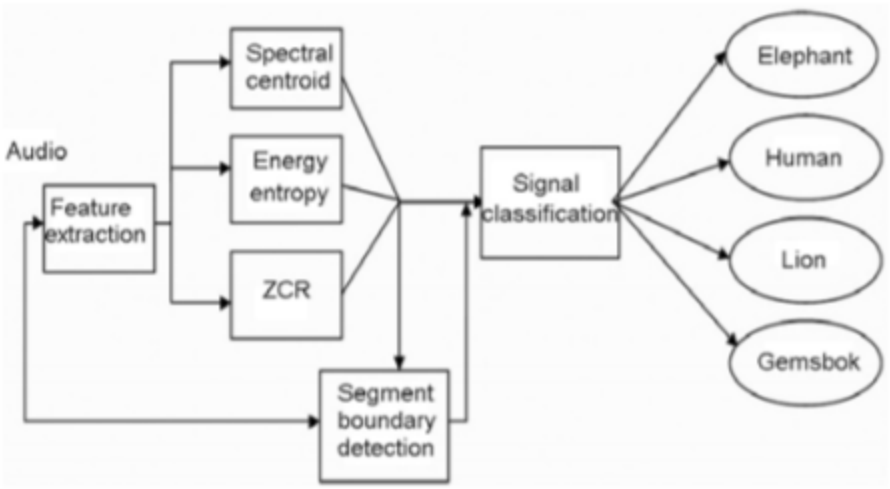

Start
Image captured by Camera
Image processing and classification
If animal found?
No
No action taken
Yes
Applying repellent technique (Buzzer)
Sending alert message
stop

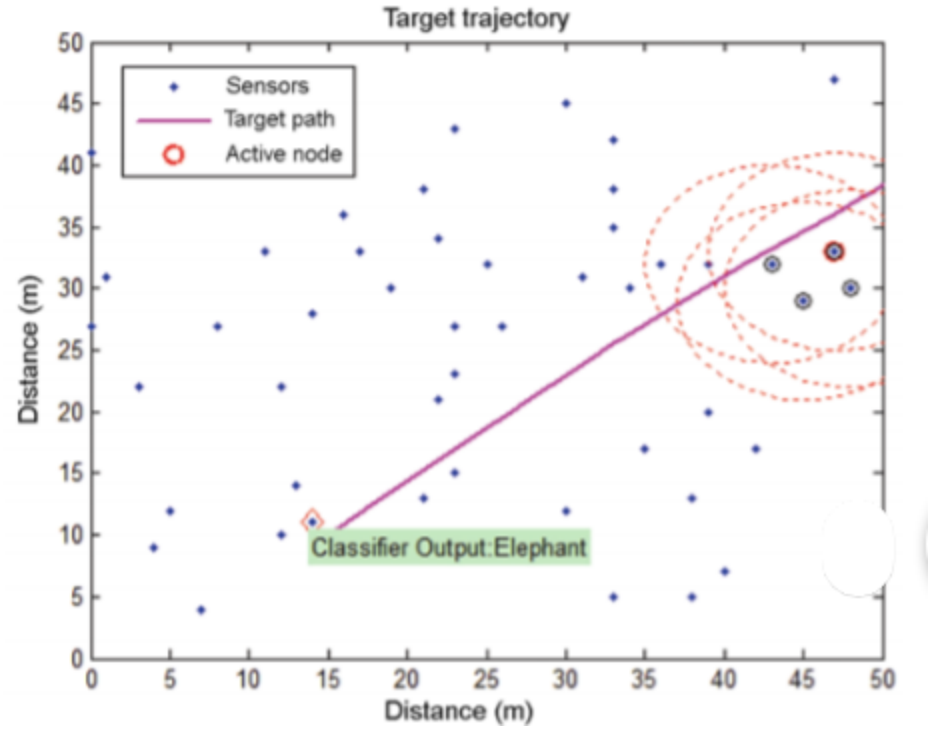
6.RESULT

As the first step datasets are collected. We know that the image taken from the cameras in the for est contain animal coming in different orientation. So the system should detect animal in any orientation. Data augmentation is performed for each images. The result of data augmentation is shown

A. Training using transfer learning In transfer learning, we take the pre-trained weights of an already trained model(one that has been trained on millions of images belonging to 1000s of classes, on several high power computers) During the training process, the images are passed through the network by applying several filters on the images at each layer. The activations from the final layer are used to find out which class the image belongs to. Here MobileNet is used as the pretrained model. The filters on the first few layers of the MobileNet learn to recognize colors and certain horizontal and vertical lines. The next few layers slowly learn to recognize trivial shapes using the lines and colors learnt in the previous layers. Then the next layers learn to recognize textures, then parts of objects like legs, eyes, nose, tail etc. Finally the filters in the last layers get activated by whole objects like elephant, tiger etc. By using a pretrained network to do transfer learning, we are simply adding a few dense layers at the end of the pretrained network and learning what combination of these already learnt features help in recognizing the objects in our new dataset. Hence we are training only a few dense layers. Furthermore, we are using a combination of these already learnt trivial features to recognize new objects. All this helps in making the training process very fast and require very less training data compared to training a CNN from scratch. To perform The training first data is stored in a particular format(size: 224 × 224 × 3, jpeg format) in order to be fed into the network to train. Then Created a data folder, inside that data folder, created folder for each class of data containing the corresponding images. The names of the folders are the names of their respective classes. Here created seven such folder , which contain seven wild animal namely elephants, lion, leopard, tiger, pig, fox and bear. Imported the pre-trained model and added dense layers. For that, First

loaded the dependencies. Then imported the pre-trained MobileNet model. The Mobilenet (trained on the imagenet dataset for a thousand classes) will have a last layer consisting of 1000 neurons (one for each class). We want as many neurons in the last layer of the network as the number of classes we wish to identify. So we discard the 1000 neuron layer and add our own last layer for the network. This was done by setting (IncludeTop=False) when importing the model. Here we need to identify seven classes of animal so we need seven neurons in the final layer. This is step 1 of the process. Imported the MobileNet model without its last layer and added a few dense layers so that our model can learn more complex functions. The dense layers has ReLU activation function and the last layer, which contains seven neurons has the softmax activation. Now that we had our model, as we will be using the pretrained weights, that our model has been trained on (imagenet dataset), we have to set all the weights to be non-trainable. We will only be training the last Dense layers that we have made previously. Now we move onto Step 2 of the process, In this step loaded the training,We just have to specify the path to our training data and it automatically sends the data for training, in batches. Next we move onto Step 3, training the model on the dataset. first compiled the model that we have made, and then trained our model with our labeled dataset. With this, we will have trained a model. The trained model can then be used to predict which class a new unseen image belongs to The result obtained using the created model is shown in It can detect the presence of any animal even if the background is changed. the Fig.7 shows the detection of elephant with different back


Audio
Feature extraction
Spectral centroid
Energy entropy
ZCR
Segment boundary detection
Signal classification
Elephant
Human
Lion
Gemsbok


Target trajectory
Sensors
Target path
Active node
Classifier Output:Elephant
Distance (m)
Distance (m)
0
5
10
15
20
25
30
35
40
45
50

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- ➤ Features are automatically deduced and optimally tuned for desired outcome. Features are not required to be extracted ahead of time. This avoids time consuming machine learning techniques.
- ➤ Robustness to natural variations in the data is automatically learned. The same neural network based approach can be applied to many different applications and data types.
- ➤ Massive parallel computations can be performed using GPUs and are scalable for large volumes of data. Moreover it delivers better performance results when amount of data are huge.
- ➤ The deep learning architecture is flexible to be adapted to new problems in the future.

DISADVANTAGES

- ➤ Screen is difficult to read in low light or bright sunlight
- ➤ Synchronising time consuming and awkward
- ➤ Keeping track of cameras and images (no meta-data)
- ➤ Losing settings during transit
- ➤ Walk-by test requires downloading image on laptop in the field

- .Excessive use of flash and frequent triggering (mostly generating false positives)
- Snow/sleet and ice build-up and condensation on lens

- Loss of clock synchrony between cameras, with rate of divergence changing over deployment period
- Data management
- Loss of meaningful date-time stamps

8.APPLICATIONS

- Automatically Adding Sounds To Silent Movies
- Automatic Machine Translation
- Object Classification and Detection in Photographs
- Automatic Handwriting Generation
- .Character Text Generation.
- Image Caption Generation.
- .Automatic Game Playing.

9.CONCLUSION

The paper discusses in detail all advances in the area of automated wildlife monitoring . Animal detection methods are very useful for many real time applications. Automated animal detection, localization, and species recognition lie in the heart of automated camera-trap image analysis. Animal detection approaches are helpful to prevent dangerous situations caused by the wild animals in residential area. Detection of animal in the border region helps the people living in the border region to take safety

precaution. Hence this system ensure the security of both wild animal and human beings in the border region.

10.FUTURE SCOPE

This work can be further extended by sending an alert in the form of a message when the animal is detected to the nearby forest office. Furthermore it can be used to reduce human wildlife conflict and also animal accidents. References

11.BIBLIOGRAPHY

- Baraldi, A., & Parmiggiani, F. (1995). An investigation of the textural characteristics associated with gray level cooccurrence matrix statistical parameters. Geoscience and Remote Sensing, IEEE Transactions on, 33, 293–304. https://doi.org/10.1109/36.377929
- Barnich, O., & Van Droogenbroeck, M. (2011). Vibe: A universal back- ground subtraction algorithm for video sequences. IEEE Transactions on Image Processing, 20, 1709–1724. https://doi.org/10.1109/ TIP.2010.2101613
- Bowler, M. T., Tobler, M. W., Endress, B. A., Gilmore, M. P., & Anderson, M. J. (2016). Estimating mammalian species richness and occupancy in tropical forest canopies with arboreal camera traps. Remote Sensing in Ecology and Conservation, 3, 146–157. Candès, E. J., Li, X., Ma, Y., & Wright, J. (2011). Robust principal com- ponent analysis? Journal of the ACM, 58, 11–37. https://doi.org/10.1145/1970392.1970395
- Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 1, pp. 886– 893. IEEE. Dong, P., Wang, S., Xia, Y., Liang, D., & Feng, D. D. (2016). Foreground detection with simultaneous dictionary learning and historical pixel maintenance. IEEE Transactions on Image Processing, 25, 5035–5049. https://doi.org/10.1109/TIP.2016.2598680
- Fei-Fei, L., & Perona, P. (2005). A bayesian hierarchical model for learning natural

scene categories. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 2, pp. 524–531. IEEE. Gregory, T., Carrasco Rueda, F., Deichmann, J., Kolowski, J., & Alonso, A. (2014). Arboreal camera trapping: Taking a proven method to new heights. Methods in Ecology and Evolution, 5, 443–451. https://doi. org/10.1111/2041-210X.12177

- He, J., Balzano, L., & Szlam, A. (2012). Incremental gradient on the grass- mannian for online foreground and background separation in sub- sampled video. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pp. 1568–1575. IEEE. He, Z., Kays, R., Zhang, Z., Ning, G., Huang, C., Han, T. X., … McShea, W. (2016). Visual informatics tools for supporting large-scale col- laborative wildlife monitoring with citizen scientists. IEEE Circuits and Systems Magazine, 16, 73–86. https://doi.org/10.1109/ MCAS.2015.2510200
- Huang, H. C., Hsieh, C. T., & Yeh, C. H. (2015). An indoor obstacle detec- tion system using depth information and region growth. Sensors, 15, 27116–27141. https://doi.org/10.3390/s151027116 Kays,
- R. (2016). Candid creatures: How camera traps reveal the mysteries of nature. Baltimore, MD: JHU Press. Kays, R., Parsons, A. W., Baker, M. C., Kalies, E. L., Forrester, T., Costello, R., … McShea, W. J. (2016). Does hunting or hiking affect wildlife communities in protected areas? Journal of Applied Ecology, 54, 242–252. https://doi.org/10.1111/1365-2664.12700
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classifi- cation with deep convolutional neural networks. Advances in Neural Information Processing Systems, 25, 1097–1105. Lee, C. Y., Xie, S., Gallagher, P., Zhang, Z., & Tu, Z. (2014). Deeply-super- vised nets. arXiv preprint arXiv:14095185. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). Ssd: Single shot multibox detector. European conference on computer vision (pp. 21–37). Cham, Switzerland: Springer. McShea, W. J., Forrester, T., Costello, R., He, Z., & Kays, R. (2016). Volunteer-run cameras as distributed sensors for macrosystem mam- mal research. Landscape Ecology, 31, 55–66. https://doi.org/10.1007/ -9

APPENDIX

HTML

```
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Wild animals Recognition </title>
  <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
rel="stylesheet">
  <script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
  <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
  <script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
  <link href="{{ url_for('static', filename='css/main.css') }}" rel="stylesheet">
        <style>
        .bg-dark {
                background-color: #42678c!important;
        }
        #result {
                color: #0a1c4ed1;
        }
        </style>
</head>

<body>
  <nav class="navbar navbar-dark bg-dark">
    <div class="container">
      <a class="navbar-brand" href="#">Intelligent Alert System for Forest Tribal People
Using CNN</a>
    </div>
```

```
  </nav>
  <div class="container">
    <div id="content" style="margin-top:2em">
                <div class="container">
                 <div class="row">
                        <div class="col-sm-6 bd" >
                         <h3>ALERT SYESTEM </h3>
                         <br>
                         <p>Having accurate, detailed, and up-to-date information about the
location and behavior of animals in the wild would revolutionize our
ability to study and conserve ecosystems. We investigate the ability to automatically,
accurately, and inexpensively collect such data,
which could transform many fields of biology, ecology, and zoology
into “big data” sciences. Motion sensor “camera traps” enable collecting wildlife pictures
inexpensively, unobtrusively, and frequently.
However, extracting information from these pictures remains an expensive, time-consuming,
manual task. We demonstrate that such information can be automatically extracted by deep
learning, a cuttingedge type of artificial intelligence. We train deep convolutional neural
networks to identify, count, and describe the behaviors of 48
species in the 3.2-million-image Snapshot Serengeti dataset. Our
deep neural networks automatically identify animals with over 93.8%
accuracy, and we expect that number to improve rapidly in years
to come. More importantly, if our system classifies only images it
is confident about, our system can automate animal identification
for 99.3% of the data while still performing at the same 96.6% accuracy as that of
crowdsourced teams of human volunteers, saving
more than 8.4 years (at 40 hours per week) of human labeling effort</p>
          <img
src="https://e360.yale.edu/assets/site/_1500x1500_fit_center-center_80/AcousticMonitor_w
eb.jpg", style="height:450px"class="img-rounded" alt="Gesture">
          </div>
                        <div class="col-sm-6">
```

```
                    <div>
                        <h4>Please uploand an image</h4>
                <form action = "http://localhost:5000/predict" id="upload-file"
method="post" enctype="multipart/form-data">
                    <label for="imageUpload" class="upload-label">
                        Choose...
                    </label>
                    <input type="file" name="image" id="imageUpload" accept=".png,
.jpg, .jpeg">
                </form>


                <div class="image-section" style="display:none;">
                    <div class="img-preview">
                        <div id="imagePreview">
                        </div>
                    </div>
                    <div>
                        <button type="button" class="btn btn-info btn-lg "
id="btn-predict">SUBMIT</button>
                    </div>
                </div>

                <div class="loader" style="display:none;"></div>

                <h3>
                    <span id="result"> </span>
                </h3>

            </div>
                </div>
```

```
                </div>
                </div>
                </div>
    </div>
</body>

<footer>
    <script src="{{ url_for('static', filename='js/main.js') }}" type="text/javascript"></script>
</footer>

</html>
```

APP.PY

```
import numpy as np
import os
from keras.models import load_model
from keras.preprocessing import image
import tensorflow as tf
global graph
graph = tf.get_default_graph()
from flask import Flask , request, render_template
from werkzeug.utils import secure_filename
from gevent.pywsgi import WSGIServer

app = Flask(__name__)
model = load_model("mymodel.h5")

@app.route('/')
def index():
    return render_template('base.html')
```

```
@app.route('/predict',methods = ['GET','POST'])
def upload():
    if request.method == 'POST':
        f = request.files['image']
        print("current path")
        basepath = os.path.dirname(__file__)
        print("current path", basepath)
        filepath = os.path.join(basepath,'uploads',f.filename)
        print("upload folder is ", filepath)
        f.save(filepath)

        img = image.load_img(filepath,target_size = (64,64))
        x = image.img_to_array(img)
        x = np.expand_dims(x,axis =0)

        with graph.as_default():
            preds = model.predict_classes(x)

            print("prediction",preds)

        index = ['Humans','Wild animals']

        text = "the predicted is : " + index[preds[0]]

    return text
if __name__ == '__main__':
    app.run(debug = True)
```