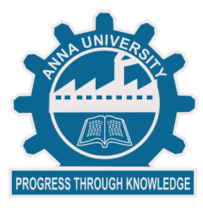
****

**CATTLE BREED IDENTIFICATION USING DEEP LEARNING**

**A PROJECT REPORT**

***Submitted by***

**SABITHA T (1805120)**

**SAMAYANJALI M (1805121)**

**YOGAPRIYA P (1805161)**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF TECHNOLOGY**

***in***

**INFORMATION TECHNOLOGY**

**SRI RAMAKRISHNA ENGINEERING COLLEGE**

[Educational Service : SNR Sons Charitable Trust]  
[Autonomous Institution, Reaccredited by NAAC with ‘A+’ Grade]  
[Approved by AICTE and Permanently Affiliated to Anna University, Chennai]  
[ISO 9001:2015 Certified and all eligible programmes Accredited by NBA]  
 VATTAMALAIPALAYAM, N.G.G.O. COLONY POST, COIMBATORE – 641 022.

**COIMBATORE-641 022**

**ANNA UNIVERSITY : CHENNAI 600 025**

**MAY 2022**

**ANNA UNIVERSITY : CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

**16IT266 PROJECT WORK**

Certified that this project report **“CATTLE BREED IDENTIFICATION USING DEEP LEARNING”** is the bonafide work of “**SABITHA T (1805120), SAMAYANJALI M (1805121), YOGAPRIYA P (1805161)”** who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE**  Dr. M. Senthamil Selvi  **HEAD OF THE DEPARTMENT**  Professor  Department of Information Technology  Sri Ramakrishna Engineering College  Coimbatore-641022 | **SIGNATURE**  Dr. J. Anitha  **SUPERVISOR**  Associate Professor  Department of Information Technology  Sri Ramakrishna Engineering College  Coimbatore-641022 |

**Submitted for the Project Viva-Voce examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We thank the almighty for his blessings on us to complete this project work successfully.

With profound sense of gratitude we thank the Management, Managing Trustee**, Thiru. D.Lakshminarayanaswamy** and Joint Managing Trustee, **Thiru. R.Sundar** for having provides us the necessary infrastructure required for the completion of our project.

With profound sense of gratitude, we sincerely thank the **Head of the Institution**, **Dr. N.R. Alamelu** for her kind patronage, which helped in pursuing the project successfully.

With immense pleasure, we express our hearty thanks to **Head of the Department**, **Dr. M. Senthamil Selvi** for her encouragement towards the completion of this project..

We thank our Academic Coordinator **Dr.K.Deepa, Associate Professor**  and Project Coordinator **Dr. M. Kalaiarasu,**  **Associate Professor,** Department of Information Technology for their encouragement towards the completion of this project.

We also thank our Project guide **Dr.J.Anitha, Associate Professor** Department of Information Technology for his guidance towards the completion of the project.

We convey our thanks to all the teaching and non-teaching staff members of our department who rendered their co-operation by all means for completion of this project.

**ABSTRACT**

Animals breeds are determined using cattle identification techniques based on their size and colour. Radio frequency, optical frequency, and ear tagging are currently used among the many alternatives available for forecasting individual cattle breeds. An image of the cattle will be taken to get the prominent features of the front, rear and side view and these extracted features will be taken to identify each individual cattle with their information. Each image dataset will be processed and according to the processed image the cattle breed will be determined. Each cattle image will be pre-processed and after pre-processing of the images they will be categorized into their respective breed. A separate category of rare breeds of cattle’s will be categorized using the pre-processed image dataset of the cattle breeds. CNN deep learning method is used in this work to recognize the face and different shapes in the whole body of the cattle image. CNN was used to extract features from a cattle image dataset. The dataset consists of 150 images was taken then identify 13 categories of cattle breeds from different regions in India. A CNN network was used to extract

features from a rear-view cattle image dataset and these extracted features were then used to train. Our approach outperformed the framework that only uses CNN. Overall, our approach can capture extract features to improve cattle identification accuracy, enabling automated cattle identification for precision livestock farming.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page No.** |
|  | **ABSTRACT** | iv |
|  | **LIST OF FIGURES** | vi |
| **1** | **INTRODUCTION** | 01 |
| **2** | **LITERATURE SURVEY** | 03 |
| **3** | **PROJECT DESCRIPTION** | **07** |
|  | 3.1 INTRODUCTION | 07 |
|  | 3.2 BLOCK DIAGRAM | 08 |
|  | 3.3 WORKING PRINCIPLE | 09 |
|  | 3.4 MODULES |  |
|  | 3.5 MERITS AND DEMERITS | 10 |
|  | 3.6 APPLICATIONS | 11 |
| **4** | **RESULTS** | 12 |
| **5** | **CONCLUSION AND FUTURE SCOPE** | 13 |
|  | **APPENDICES** |  |
| **APPENDIX1** | **SOFTWARE DESCRIPTION** | 14 |
| **APPENDIX2** | **SOURCE CODE** | 17 |
| **APPENDIX 3** | **SCREENSHOTS** | 23 |
| **APPENDIX 4** | **PUBLICATIONS** |  |
|  | **REFERENCES** | 26 |
|  |  |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Fig No.** | **Title** | **Page No.** |
| 4.1 | Block Diagram | 14 |
| 4.2 | Accuracy | 15 |
| A3.1 | Python GUI Representation | 15 |
| A3.2 | Cattle Detection - Bargur | 18 |
| A3.3 | Cattle Detection - Gir | 19 |
| A3.4 | Cattle Detection - Hallikar | 20 |
| A3.5 | Cattle Detection -Red Sindhi | 21 |
| A3.6 | Cattle Detection - Kangayam | 21 |
| A3.7 | Cattle Detection -Deoni | 22 |

**LIST OF ABBREVATIONS**

|  |  |
| --- | --- |
| CNN | Convolutional Neural Network |
| SVM | Support Vector Machine |
| KNN | K Nearest Neighbor |
| RFID | Radio Frequency Identification |
| SFTA | Segmentation-based Fractal Texture Analysis |
| GUI | Graphical User Interface |

**LIST OF TABLES**

|  |  |
| --- | --- |
| CNN | Convolutional Neural Network |

**CHAPTER 1**

**INTRODUCTION**

Dairy farming is one of the key industries that employs both landowners and landless laborer. Animal breeds are intraspecific groups that share specific qualities that separate them from other breeds in today's globe. A farmer's breed selection is influenced by a number of elements, including milk, environmental conditions, and so on. Crossbreeds are becoming increasingly popular, and certain varieties are near extinction. According to research by the Food and Agriculture Organization (FAO), 26% of livestock are endangered.

Sensors/RFID (Radio Frequency Identification) are one of the methods used to recognize their behaviors, which were previously employed by conventional methods, however the attachment of sensors to the animal may cause stress and harm. The cameras were mostly used to recognize cattle movement on cattle farms, however it's difficult to distinguish cattle when they look alike, especially black or brown cattle, and the sensors employed can quickly fail or be harmed due to collisions between cattle sensors. Ear tagging is one of the most commonly used ways.

The practice of successfully recognizing individuals using a unique identity or biometric traits is known as cattle identification. Individual cow identification is required for automated study of animal actions and productivity in precision livestock management. On-animal sensors such as ear tags, collars, and radio frequency identification modules are commonly used in traditional cattle identification systems, which entail costs and may also burden cattle. Furthermore, in severe outdoor conditions, RFID tags or sensors are prone to loss or damage. As a result, a more reliable and accurate livestock identification system is desired.

Deep learning networks with automatic feature extraction and powerful image representation capability have been widely used in the field of object detection, visual recognition and image segmentation. As a result, there has been recent interest in the use of deep learning for cattle feature extraction and identification of individual animals. In approaches, deep learning models such as Convolutional Neural Networks (CNN) are utilized to extract high-dimensional visual features in a spatial domain from images, with these extracted features then being used to identify animals through a classifier layer.

**CHAPTER 2**

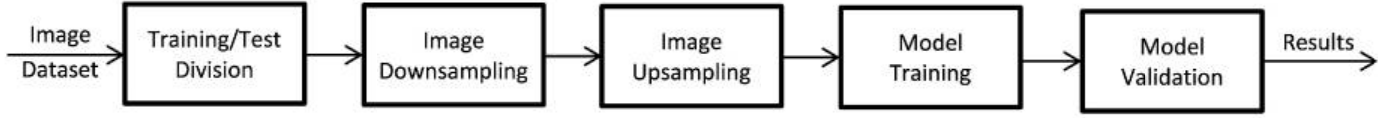
**LITERATURE SURVEY**

Various researches have been done for Cattle Breed Identification using Deep Learning. This research is done prior to taking up the project and understanding the various methods that were used previously.

**2.1 Detection of Cattle in UAV Images Using Deep Learning**

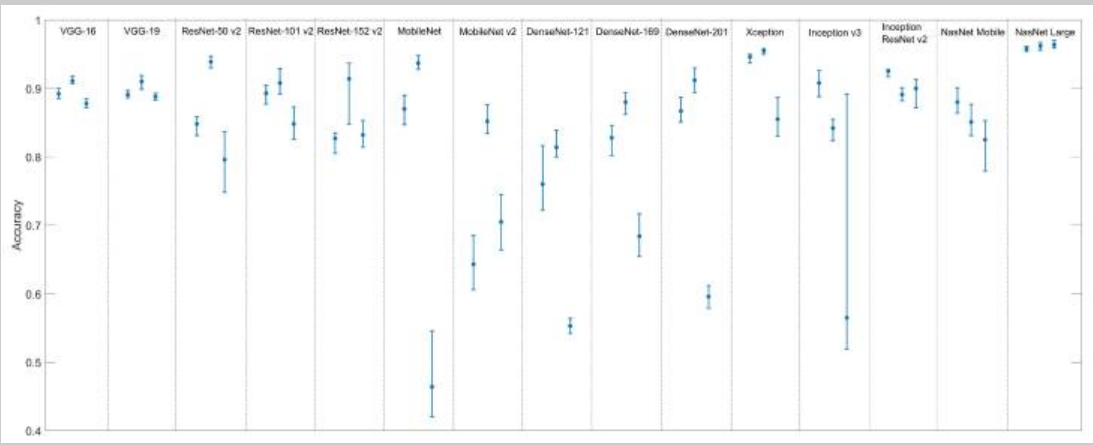
**Authors : Jayme G A Barbedo, Luciano Koenigkan, Thiago Santos**

In this project, the objectives of this study is to determine the highest possible accuracy that could be achieved in the detection of cattle and to determine the most accurate CNN architecture for this specific problem. The experiments involved 1853 images containing 8629 samples of animals, and 15 different CNN architectures were tested. A total of 900 models were trained (15 CNN architectures × 3 spatial resolutions × 2 datasets × 10-fold cross validation), allowing for a deep analysis of the several aspects that impact the detection of cattle using aerial images captured using UAVs. Results revealed that many CNN architectures are robust enough to reliably detect animals in aerial images even under far from ideal conditions, indicating the viability of using UAVs for cattle identification. The following figure shows the basic workflow used to train each of the models tested in this approach.

****

**Fig 2.1.1 Detection of cattle in UAV images – Flow Diagram**

The result is shown below

****

Range of accuracies obtained for each CNN architecture. The three bars associated to each architecture correspond to input sizes of 224x224 (left), 112x112 (middle) and 56x56 (right). The circle in each bar represents the average accuracy, and the bottom and top extremities represents the lowest and highest accuracies observed during the application of the 10-fold-cross-validation.

**2.2 Face Recognition for Cattle using SURF**

**Authors: Santhosh Kumar, Shrikhant Tiwari, Sanjay Kumar Singh**

The algorithm is galvanized by the observation that cattle have rich skin texture and distinct facial features. The steps involved in this approach is , the pixel information in the cattle face image changes with different body changes, pose variations and illumination condition. In the cattle face images, there are numerous poses and illumination present in the face images of some cattle in the database is very low. The effects of such artefacts can be mitigated by using a Gaussian method.

****

Where N is the number of Gaussian pyramid layers.

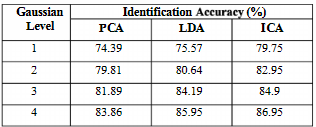
So, the steps are:

1. SURF descriptors area unit deployed to detect on the original image. Descriptor providers scale and rotation invariant data from the lowest level (G1) of Gaussian pyramid.
2. Moving down the Gaussian pyramid, the size of image decreases and it cannot get sufficient number of interest points in the images at Gaussian pyramid level 1, level 2, level 3 and level 4. Therefore, CLBP is applied at only level 1 and level 2 and to confine the more discriminating texture cattle face image information from these Gaussian pyramid levels.
3. In this approach of cattle face recognition, Chi-Square distance is employed to measure the unsimilarity between the corresponding levels of Gaussian pyramid. Suppose S0,S1,S2 be the min-max face image normalized scores for each Gaussian pyramid level.
4. Finally weighted sum rule is applied to combine the 3 match scores.

For experimental result based performance evaluation, the database of cattle face (images) was segmented into two parts: (i) Training and (ii) Testing. The three appearance based algorithms used for comparison are:

1. Principal Component Analysis (PCA)
2. Linear Discriminative Analysis (LDA)
3. Independent Component Analysis (ICA)

The following figure shows the Identification accuracy of the algorithms

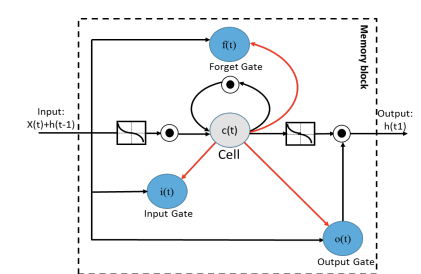
****

**Fig 2.2.1 Identification Accuracy**

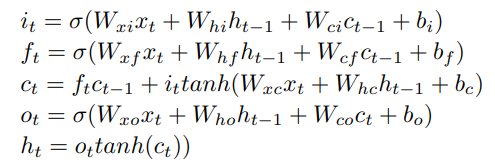
**2.3 Individual Cattle Identification using LSTM**

**Authors: Yongliang Qiao, Daobilige Su, Cameron Clark, Salah Sukkarieh**

The approach is a deep learning based framework for beef cattle identification using video data. LSTM is a popular network for spatial-time data preprocessing with strong abilities to learn and remember over long sequences of input data. To make the most of the video contents, one should consider the visual aspects and also characterize the object appearances as well as the motion present within the data. As such, after CNN features were extracted image by image from each video, we trained the LSTM model to identify the cattle video sequences.



As illustrated in the above diagram, a common LSTM unit consist of a cell, an input gate, an output gate and a forget gate. At every time step t, given the input data x, the computation of the hidden value ht of an LSTM cell was updated as follows:



where σ is the sigmoid function; tanh represents the hyperbolic tangent activation function; i, f, o and c are the input gate, forget gate, output gate and cell activation vectors, respectively; h is the hidden vector; b denotes bias vectors and matrix W is the connection weight between two units.

**CHAPTER 3**

**PROJECT DESCRIPTION**

**3.1 INTRODUCTION**

Cattle identification is the process of accurately recognizing individuals via a unique identifier or biometric features. In precision livestock management, individual cattle identification is a prerequisite for automated analysis of animal activities and productivity. Classical cattle identification methods typically adopt on-animal sensors such as ear-tags, collars, and radio frequency identification modules, which incur costs and may also burden cattle. In addition, these tags or sensors are prone to loss or damage in harsh outdoor environments. Therefore, a more robust cattle identification system of high accuracy is desirable.

Deep learning networks with automatic feature extraction and powerful image representation capability have been widely used in the field of object detection, visual recognition and image segmentation. As a result, there has been recent interest in the use of deep learning for cattle feature extraction and identification of individual animals. In approaches, deep learning models such as Convolutional Neural Networks (CNN) are utilized to extract high-dimensional visual features in a spatial domain from images, with these extracted features then being used to identify animals through a classifier layer.

* 1. **MODULES**

1. Collection of Dataset
2. Importing The Dataset and Pre-Processing
3. Training
4. Classification

**3.2.1 Collection of Dataset**

The Dataset is collected from several images around 150 different images of cattle was captured and it consists of 13 categories of cattle breeds from different regions in India.

**3.2.2** **Importing the Dataset and Pre-processing**

The dataset was loaded into the GUI. The undesirable parts of the image are deleted during pre-processing. The cattles are identified using the CNN approach.

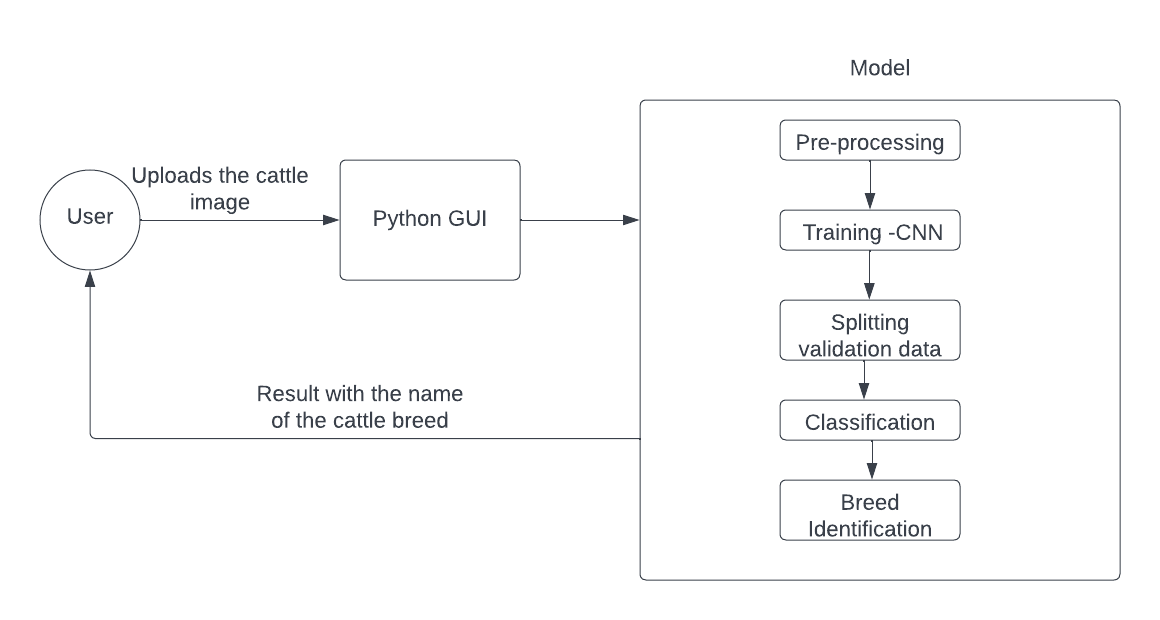
* + 1. **Training**

After pre-processing, the dataset is trained using convolutional architecture is utilized to determine a specific area or region directly. This approach is mostly used to segment image pixel values. After the cattle image has been transformed to grey scale, the image is classified into its specific number of classes.

**3.2.4 Classification**

CNN is the most commonly utilised domain that provides the best level of accuracy and classify the image using the CNN algorithm after extracting the features because CNN uses minimum knowledge of the training data. To trained the CNN model. Each method is carried out for individual cow, and the breed is identified and detected as a result.

* 1. **BLOCK DIAGRAM**



**Fig 3.3.1 BLOCK DIAGRAM**

**3.4 METHODOLOGY**

**3.4.1 INPUT IMAGES**

This is the first step in the cattle identification system, in which over 150 images of various cows were taken in various locations of India.

**3.4.2 PRE-PROCESSING**

The quality of the images was diverse due to the multiple cameras used to collect the images, this step was done after the image was loaded to the GUI as a through preparation of the images. The resolution of the photos was first increased to 247\*247 pixels throughout this process. The image’s interrupts are removed during pre-processing. The CNN method is used to increase visual contrast.

* + 1. **ALGORITHM - CONVOLUTIONAL NEURAL NETWORK**

In deep learning, a convolutional neural network (CNN) is a class of

deep neural networks, most commonly applied to analyze visual imagery. Convolutional Neural Networks are composed of multiple layers of artificial neurons. The first layer usually extracts basic features such as horizontal or diagonal edges. The output is passed on to the next layer which detects more complex features such as corners or combinational edges.

**Architecture of the model**

Initially, the convolution neural network structure was created

taking into account the structural features of some parts of the human brain responsible for vision.

Three mechanisms are laid into the foundation for the development of such networks:

- local perception;

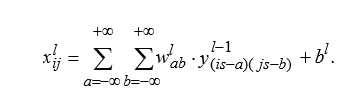
- building a set of layers in the form of characteristics maps (shared weights)

- sub-sampling.

In accordance with these arrangements, these layers are used to build a convolutional neural network:

**Convolution layer**

Convolution equation for the l -th ) (l =1,..., L network layer has the following form

. 

Convolution preserves spatial relationships between the pixels. Each convolutional layer is followed by subsampling or computational layer that is serving to reduce the dimension of the image by averaging the values of the local output neurons.

**Max Pooling Layer**

Subsampling layer zooms planes by local averaging of the output neurons values. Thus, the hierarchical organization is achieved. Subsequent layers are extracted more common characteristics that less depend on the image distortion.

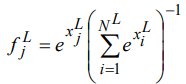
The difference between a subsampling layer and a convolution layer is that in the convolution layer regions of the neighboring neurons overlap, which does not occur in the subsampling layer.

Pooling layer operates independently of the input data depth and scales the spatial volume by using a maximum function. The architecture of the convolution network assumes that the presence of a sign is more important than information about its location. Therefore, the maximum one is selected from several neighboring neurons in the feature map and its value is considered as a single neuron in the feature map of lower dimension.

In addition to maximum subsampling, pooling layers can perform other functions, such as averaging subsampling or even L2-normalized subsampling.

**Activation Layer**

On these layers within the network, a nonlinear activation function f (•) is applied to all input values and the result is sent to the output. Thus, the activation layer does not change the input resolution. Usually due to significant positive properties for hidden layers, ReLu (ReLU (x) = max (0; x)) and its various modifications (Leaky ReLU, Parametric ReLU, Randomized ReLU) are used.

****

SoftMax function is used for fully connected layer.

**Dropout layer**

Different regularization techniques are used to avoid network retraining. Dropout is a simple and effective regularization method and consists in the fact that in the process of training a network, a subnet is randomly allocated from its aggregate topology, i.e., part of the neurons is turned off from the process, and the next update of the scales occurs only within the allocated subnet. Thus, only weights of remaining neurons are changed.

Each neuron is excluded from the total network with a certain probability, which is called the dropout rate. This layer reduces the time of one training epoch due to the smaller number of optimized parameters, and also allows to better deal with retraining of the network compared to standard regularization methods.

**Flatten Layer**

Flatten layer is used to convert the data into a 1-dimensional array for inputting it to the next layer. The flattening is used for output of the convolutional layers to create a single long feature vector. And it is connected to the final classification model, which is called a fully-connected layer. Flattening does not affect the batch size and if inputs are shaped, without a feature axis, then flattening adds an extra channel dimension.

**Dense Layer**

Dense layer is used as it is a simple layer of neurons in which each neurons receives input from all the neurons of previous layer. It feeds all outputs from the previous layer to all its neurons, each neuron provides one output to the next layer.

* 1. **MERITS AND DEMERITS**

**3.5.1 MERITS**

* It produces good accuracy depending on algorithm and environment.
* It works very efficiently and provides the required results within milliseconds.
* The executing duration is comparatively low as the model reaches good accuracy within certain epochs.
  + 1. **DEMERITS**
* Difficult to evaluate sinusitis in frontal, ethmoid, and sphenoid. In order to utilize AI based assistive software in the future, it is necessary to evaluate sinusitis at other locations as well as maxillary.
* It lacks pattern recognition and representation methods that can solve black-box in deep learning.

**CHAPTER 4**

**RESULTS AND DISCUSSION**

The implementation of the model finds and detects the cattle by using images taken of the frontal view of cattle. The model system architecture and work flow provide a useful template for similar livestock monitoring applications, such as the automated detection and identification of livestock. Therefore, the model provides the cattle breed with reference to the images and identify the category that the cattle belong to the Python GUI provides a user experience where the user can interact with the model to detect the cattle breed and use the same in the useful possible ways.

**CHAPTER 5**

**CONCLUSION AND FUTURE WORK**

We collected over 150 cattle picture datasets for our suggested system, and these photos were preprocessed to transform them to a specific dimension and reduce noise. After that, used the SIFT method to extract features, and The cattle's various body parts were extracted. Finally, we categorized CNN was used to divide the animals into 13 classes and forecast the breed. Only when the image of the cattle is visible does this proposed system work a single entire animal Identification of cattle in the future diverse backdrops with other animals in the images. Further livestock separation where the image consists Cattle of several breeds. As a result, the image pattern ensures both efficiency and animal welfare in cow farms.

**APPENDIX**

**APPENDIX 2**

**SOURCE CODE**

import numpy as np

import os

from matplotlib import pyplot as plt

import cv2

import random

import pickle

#from google.colab.patches import cv2\_imshow

from skimage.feature import hessian\_matrix, hessian\_matrix\_eigvals

file\_list = []

class\_list = []

DATADIR = "Dataset"

CATEGORIES = ["Bargur","Deoni","Gir","Hallikar","Kangayam","Red sindhi"]

IMG\_SIZE = 150

def detect\_ridges(gray, sigma=1.0):

H\_elems = hessian\_matrix(gray, sigma=sigma, order='rc')

maxima\_ridges, minima\_ridges = hessian\_matrix\_eigvals(H\_elems)

22

return maxima\_ridges, minima\_ridges

for category in CATEGORIES :

path = os.path.join(DATADIR, category)

for img in os.listdir(path):

if img not in ".ipynb\_checkpoints":

img\_array = cv2.imread(os.path.join(path, img), cv2.IMREAD\_GRAYSCALE)

print(img)

clahe = cv2.createCLAHE(clipLimit=100.0, tileGridSize=(8,8))

img\_array = clahe.apply(img\_array)

median = cv2.medianBlur(img\_array.astype('float32'), 5)

median = 255-median

ret,thresh = cv2.threshold(median.astype('uint8'),165,255,cv2.THRESH\_BINARY\_INV)

ret,thresh =

cv2.threshold(img\_array.astype('uint8'),165,255,cv2.THRESH\_BINARY\_INV)

img\_array=cv2.equalizeHist(img\_array)

img\_array = cv2.Canny(img\_array, threshold1=100, threshold2=100)

img\_array = cv2.medianBlur(img\_array,1)

new\_array = cv2.resize(thresh, (IMG\_SIZE, IMG\_SIZE))

cv2\_imshow(new\_array)

training\_data = []

def create\_training\_data():

for category in CATEGORIES :

path = os.path.join(DATADIR, category)

class\_num = CATEGORIES.index(category)

for img in os.listdir(path):

try :

img\_array = cv2.imread(os.path.join(path, img), cv2.IMREAD\_GRAYSCALE)

clahe = cv2.createCLAHE(clipLimit=100.0, tileGridSize=(8,8))

img\_array = clahe.apply(img\_array)

median = cv2.medianBlur(img\_array.astype('uint8'), 5)

median = 255-median

23

ret,thresh =

cv2.threshold(median.astype('uint8'),165,255,cv2.THRESH\_BINARY\_INV)

img\_array=cv2.equalizeHist(img\_array)

img\_array = cv2.Canny(img\_array, threshold1=30, threshold2=40)

img\_array = cv2.medianBlur(img\_array,1)

new\_array = cv2.resize(thresh, (IMG\_SIZE, IMG\_SIZE))

training\_data.append([new\_array, class\_num])

except Exception as e:

pass

create\_training\_data()

random.shuffle(training\_data)

X = [] features

y = [] labels

for features, label in training\_data:

X.append(features)

y.append(label)

X = np.array(X).reshape(-1, IMG\_SIZE, IMG\_SIZE, 1)

Creating the files containing all the information about your model

pickle\_out = open("X.pickle", "wb")

pickle.dump(X, pickle\_out)

pickle\_out.close()

pickle\_out = open("y.pickle", "wb")

pickle.dump(y, pickle\_out)

pickle\_out.close()

pickle\_in = open("X.pickle", "rb")

X = pickle.load(pickle\_in)

24

main.py

import tensorflow as tf

from tensorflow.python.keras.models import Sequential

from tensorflow.python.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D,

MaxPooling2D

import pickle

from keras.models import model\_from\_json

from keras.models import load\_model

import matplotlib.pyplot as plt

import numpy as np

X = pickle.load(open("X.pickle", "rb"))

y = pickle.load(open("y.pickle", "rb"))

# normalizing data (a pixel goes from 0 to 255)

X = X/255.0

model = Sequential()

model.add(Conv2D(32, (3, 3), input\_shape = X.shape[1:]))

model.add(Activation("relu"))

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Conv2D(64, (3, 3)))

model.add(Activation("relu"))

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Conv2D(64, (3, 3)))

model.add(Activation("relu"))

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Dropout(0.25))

model.add(Flatten())

model.add(Dense(128))

model.add(Activation("relu"))

model.add(Dense(128))

model.add(Activation("relu"))

model.add(Dense(10))

model.add(Activation("softmax"))

model.compile(loss="sparse\_categorical\_crossentropy",

optimizer="adam",

metrics=["accuracy"])

y=np.array(y)

history = model.fit(X, y, batch\_size=200, epochs=10, validation\_split=0.5)

model\_json = model.to\_json()

with open("model.json", "w") as json\_file :

json\_file.write(model\_json)

model.save\_weights("model.h5")

print("Saved model to disk")

model.save('CNN.model')

acc = history.history['accuracy']

val\_acc = history.history['val\_accuracy']

loss = history.history['loss']

val\_loss = history.history['val\_loss']

acc=np.array(acc)

val\_acc=np.array(val\_acc)

loss=np.array(loss)

val\_loss=np.array(val\_loss)

epochs\_range = range(2)

plt.figure(figsize=(15, 15))

plt.subplot(2, 2, 1)

plt.plot(epochs\_range, acc, label='Training Accuracy')

plt.plot(epochs\_range, val\_acc, label='Validation Accuracy')

plt.legend(loc='lower right')

plt.title('Training and Validation Accuracy')

plt.subplot(2, 2, 2)

plt.plot(epochs\_range, loss, label='Training Loss')

plt.plot(epochs\_range, val\_loss, label='Validation Loss')

plt.legend(loc='upper right')

plt.title('Training and Validation Loss')

plt.show()

final.py

import numpy as np

import matplotlib.pyplot as plt

import cv2

import os

import tensorflow as tf

from keras.models import load\_model

from tkinter import \*

import tkinter.messagebox

import PIL.Image

import PIL.ImageTk

from tkinter import filedialog

CATEGORIES=["Bargur","Deoni","Gir","Hallikar","Kangayam","Red sindhi"]

root = Tk()

root.title("MUZZLE BASED COW DETECTION")

root.state('zoomed')

root.configure(bg='#D3D3D3')

root.resizable(width = True, height = True)

value = StringVar()

panel = Label(root)

model = tf.keras.models.load\_model("CNN.model")

def prepare(file):

IMG\_SIZE = 150

img\_array = cv2.imread(file, cv2.IMREAD\_GRAYSCALE)

clahe = cv2.createCLAHE(clipLimit=100.0, tileGridSize=(8,8))

img\_array = clahe.apply(img\_array)

median = cv2.medianBlur(img\_array.astype('uint8'), 5)

median = 255-median

ret,thresh = cv2.threshold(median.astype('uint8'),165,255,cv2.THRESH\_BINARY\_INV)

img\_array = cv2.Canny(img\_array, threshold1=40, threshold2=70)

img\_array = cv2.medianBlur(img\_array,1)

new\_array = cv2.resize(thresh, (IMG\_SIZE, IMG\_SIZE))

return new\_array.reshape(-1, IMG\_SIZE, IMG\_SIZE, 1)

def detect(filename):

prediction = model.predict(prepare(filename))

prediction = list(prediction[0])

print(prediction)

l=CATEGORIES[prediction.index(max(prediction))]

print(CATEGORIES[prediction.index(max(prediction))])

28

value.set(CATEGORIES[prediction.index(max(prediction))])

i=int(prediction.index(max(prediction)))

def ClickAction(event=None):

filename = filedialog.askopenfilename()

img = PIL.Image.open(filename)

img = img.resize((250,250))

img = PIL.ImageTk.PhotoImage(img)

global panel

panel = Label(root, image = img)

panel.image = img

panel = panel.place(relx=0.435,rely=0.3)

detect(filename)

button = Button(root, text='CHOOSE FILE', font=(None, 18), activeforeground='red', bd=20,

bg='cyan', relief=RAISED, height=3, width=20, command=ClickAction)

button = button.place(relx=0.40, rely=0.05)

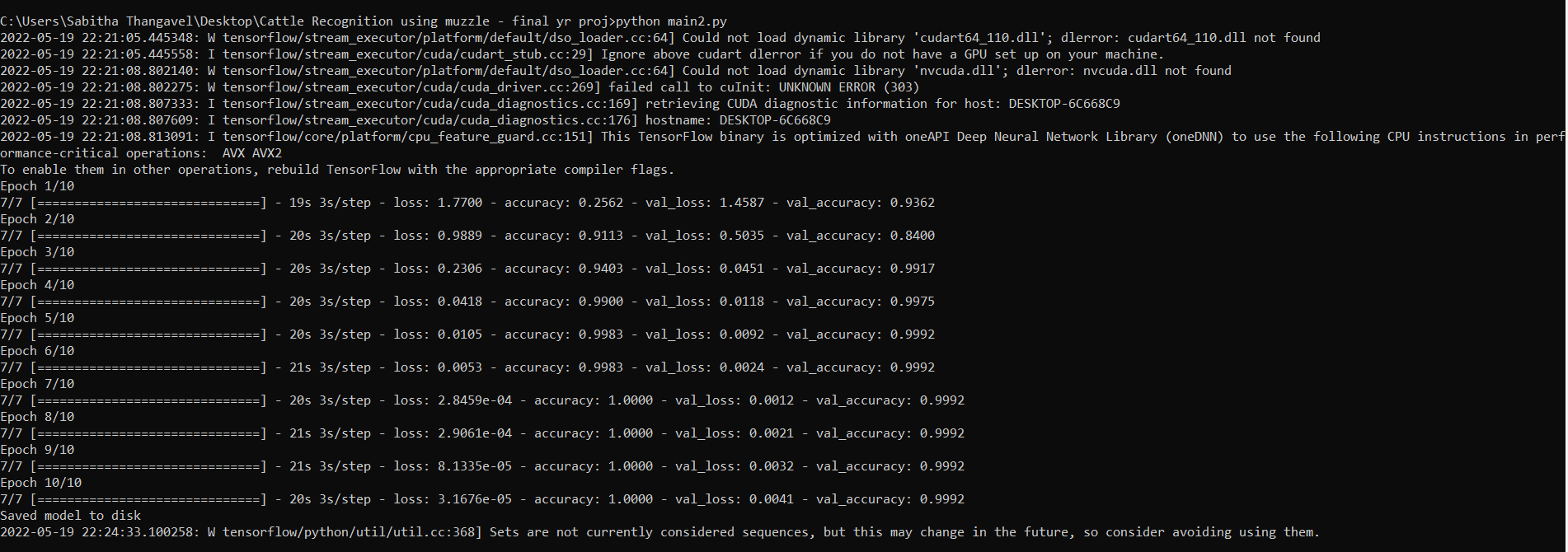
result = Label(root, textvariable=value, font=(None, 20))

result = result.place(relx=0.465,rely=0.7)

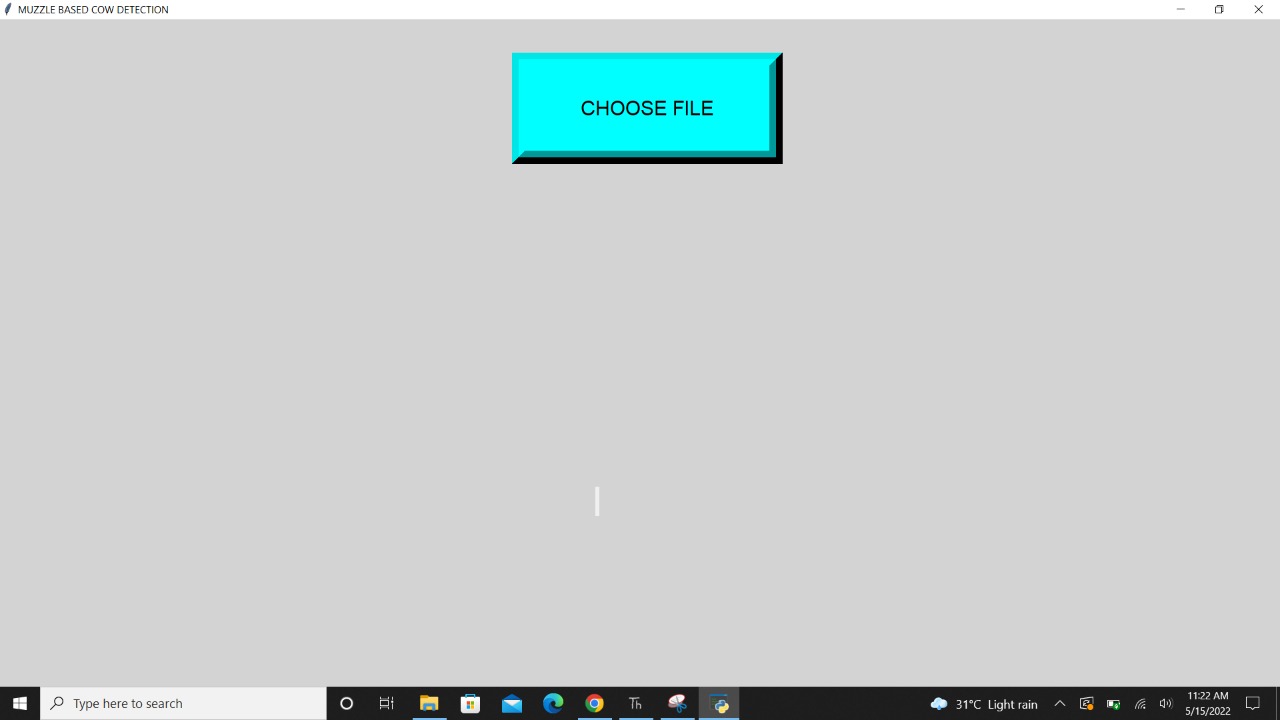
root.mainloop()

**APPENDIX**

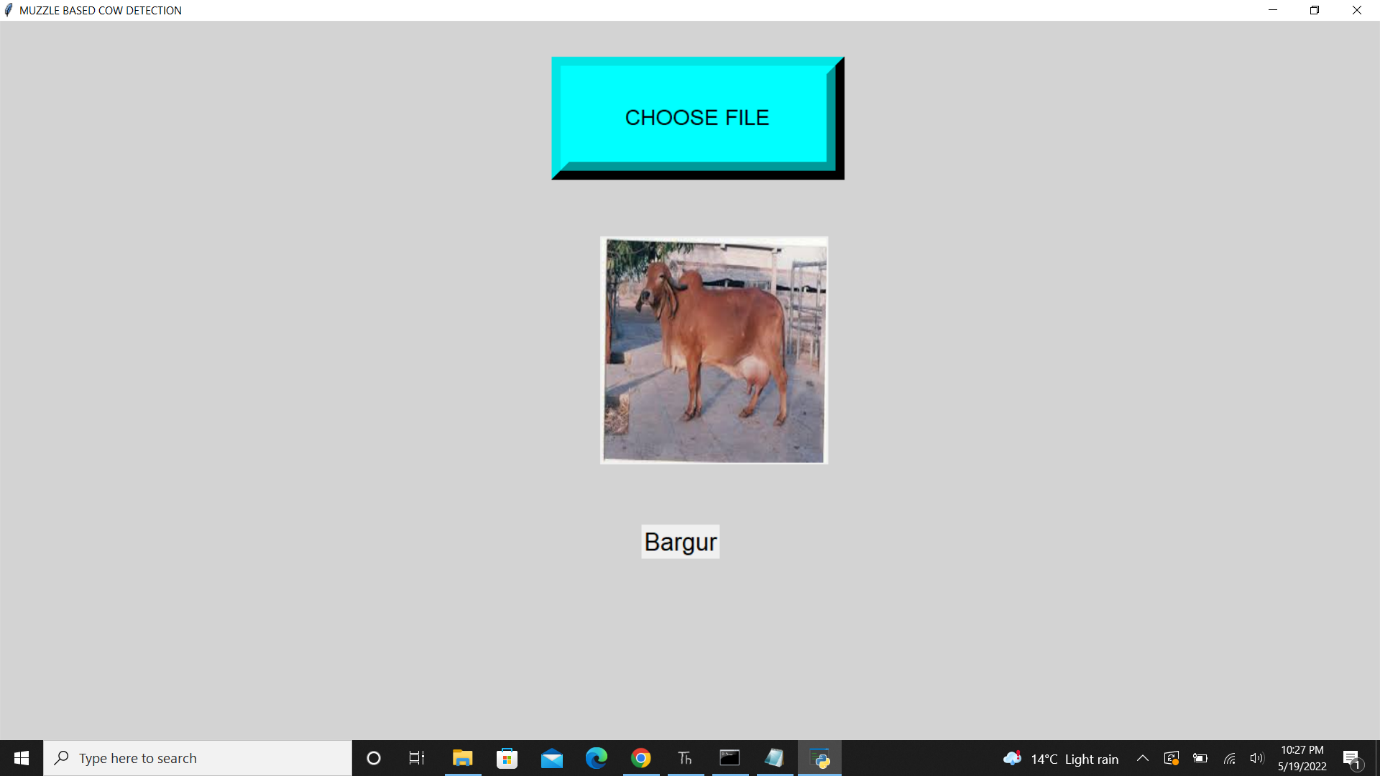
**SCREENSHOTS**



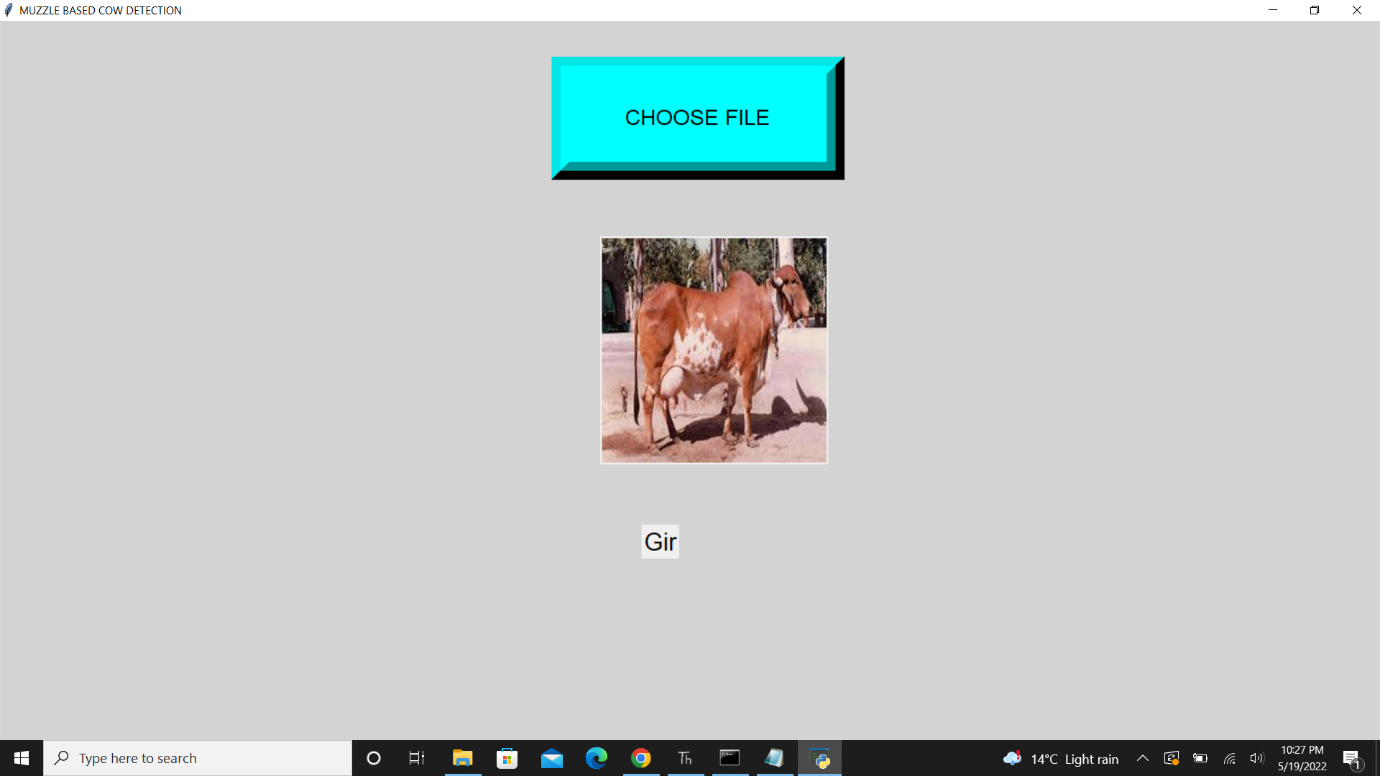
**A3.1 Accuracy**



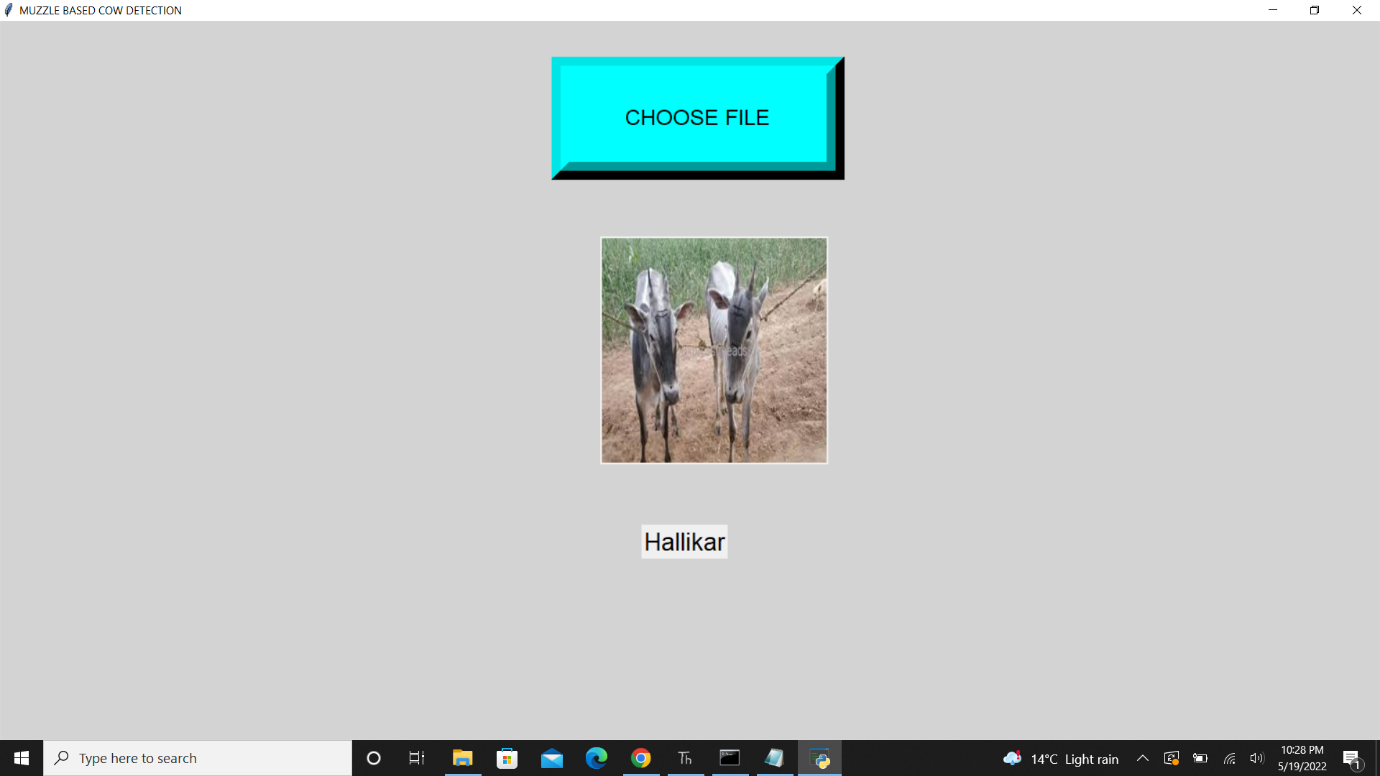
**A3.2 Python GUI Representation**



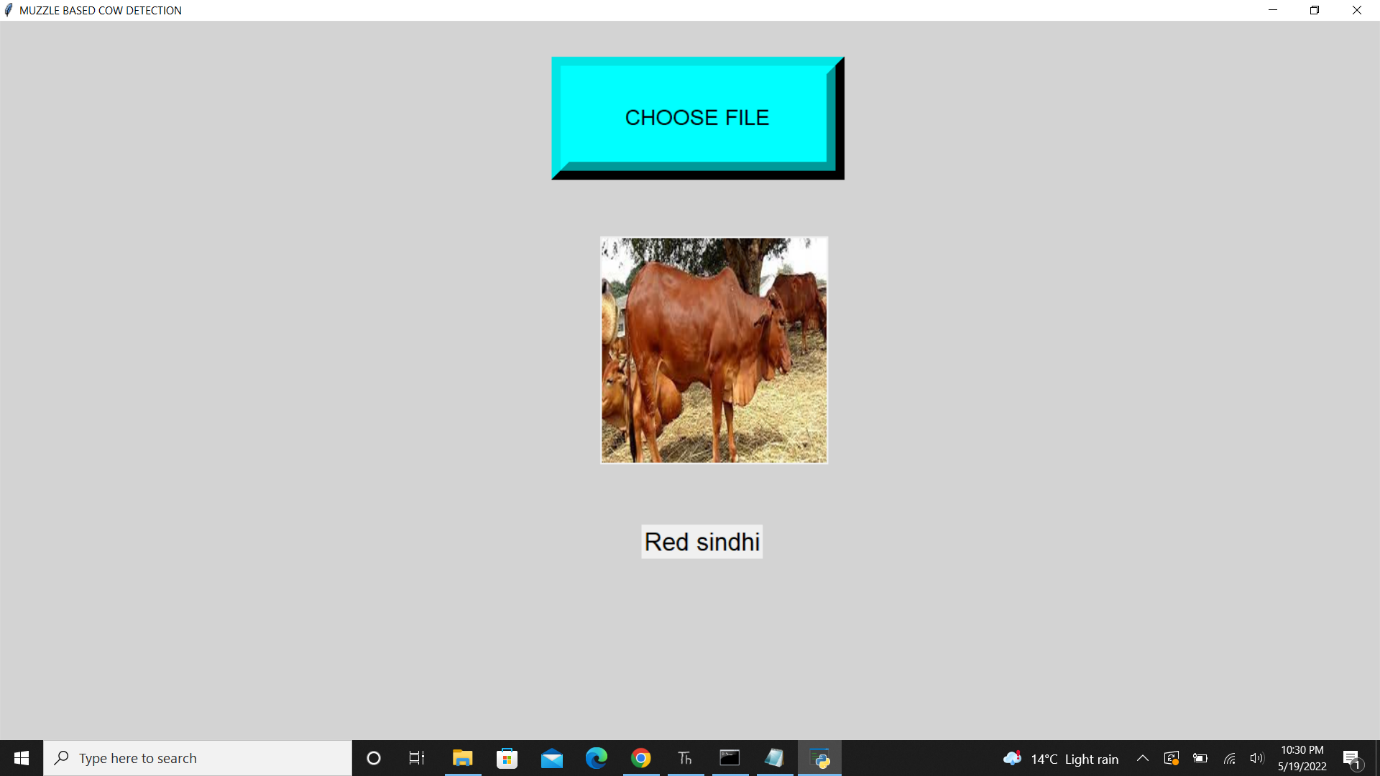
**A3.3 Cattle detection - Bargur**



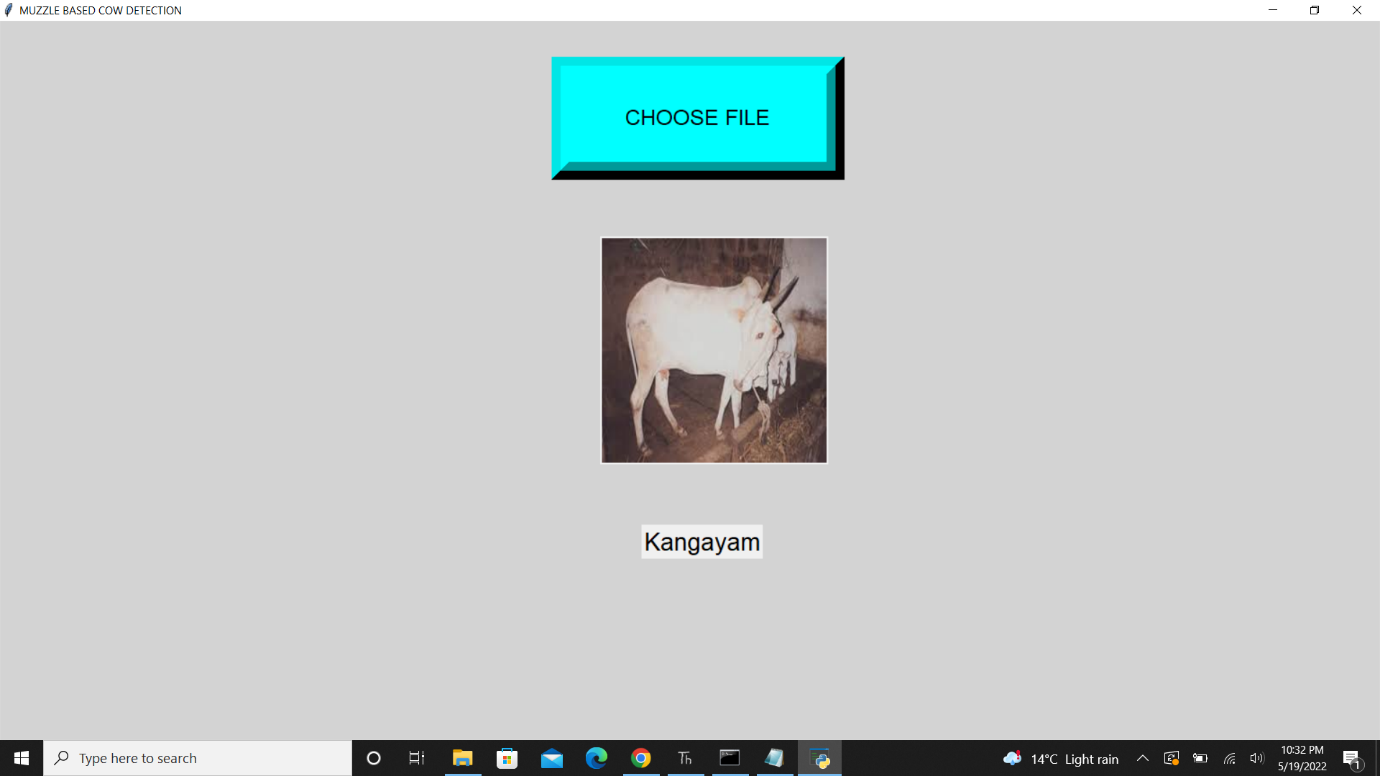
**A3.4 Cattle detection – Gir**



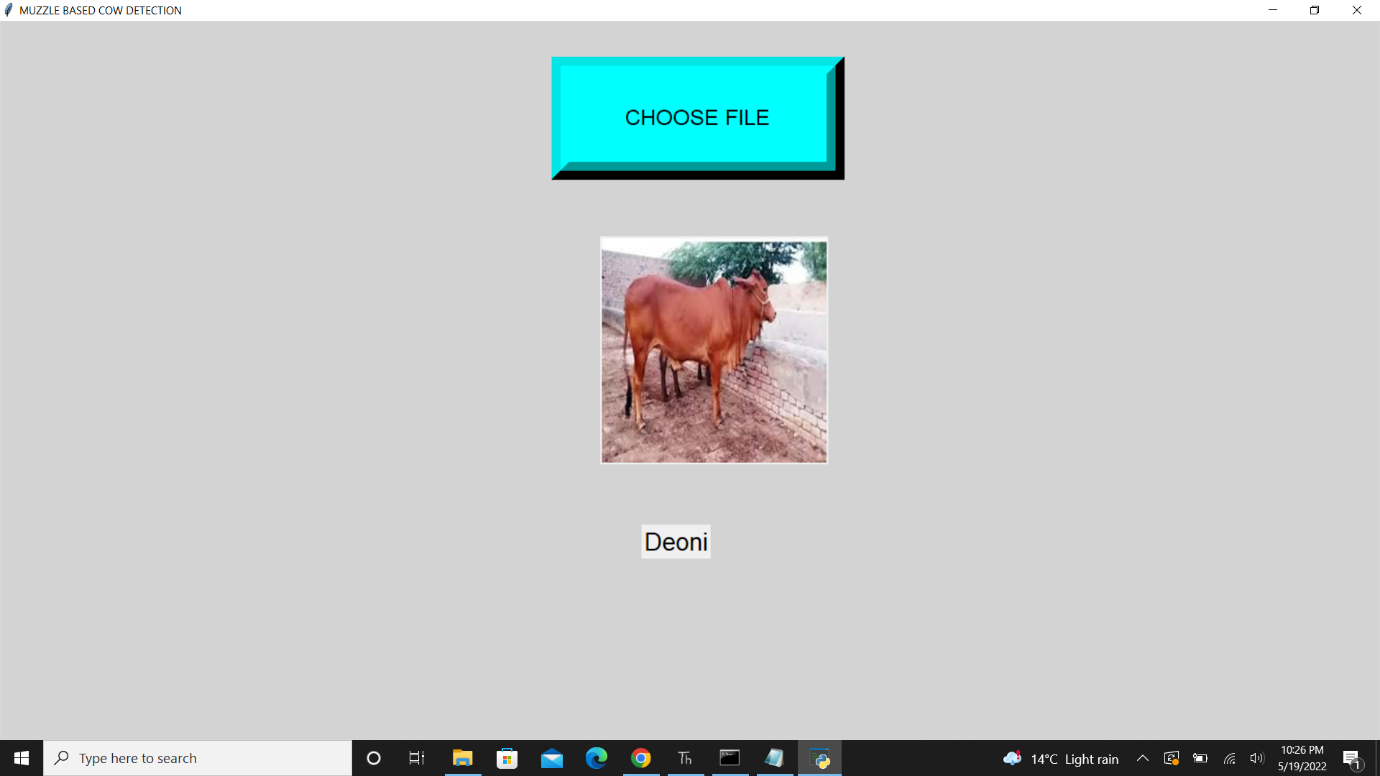
**A3.5 Cattle detection - Hallikar**



**A3.6 Cattle detection – Red Sindhi**



**A3.7 Cattle detection – Kangayam**

****

**A3.8 Cattle Detection - Deoni**

**REFERENCES**

[1] Okura, F., Ikuma, S., Makihara, Y., Muramatsu, D., Nakada, K., and Yagi, Y. (2019). RGBD video-based individual identification of dairy cows using gait and texture analyses. Comput. Electron. Agric. 165:104944. doi: 10.1016/j.compag.2019.104944.

[2] Alberto, R., C. Pablo, G.-B. Alfonso, and C. Juan. 2018. “Detection Of Cattle Using Drones and Convolutional Neural Networks.” Sensors 18 (7): 2048.

[3] Chen, Y., Z. Wang, Y. Peng, Z. Zhang, Y. Gang, and J. Sun. 2018. “Cascaded Pyramid Network for Multi-person Pose Estimation.” Paper presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City,USA.

[4] S. Kumar, S. Chandrakar, A. Panigrahi and S. K. Singh, "Muzzle point pattern recognition system using image pre-processing techniques," 2017 Fourth International Conference on Image Information Processing (ICIIP), 2017, pp. 1-6, doi: 10.1109/ICIIP.2017.8313697.

[5] W. Kusakunniran, A. Wiratsudakul, U. Chuachan, S. Kanchanapreechakorn and T. Imaromkul, "Automatic cattle identification based on fusion of texture features extracted from muzzle images," 2018 IEEE International Conference on Industrial Technology (ICIT), 2018, pp. 1484-1489, doi: 10.1109/ICIT.2018.8352400.

[6] T. T. Zin et al., "Cow Identification System using Ear Tag Recognition," 2020 IEEE 2nd Global Conference on Life Sciences and Technologies (LifeTech), 2020, pp. 65-66, doi: 10.1109/LifeTech48969.2020.1570625232.

[7] Dairy-Australia. Precision Dairy Technology Electronic Cow Identification; Dairy Australia Limited: Southbank, Australia, 2016.

[8] Crouse, D.; Jacobs, R.L.; Richardson, Z.; Klum, S.; Jain, A.; Baden, A.L.; Tecot, S.R. LemurFaceID: A face recognition system to facilitate individual identification of lemurs. BMC Zool. 2017, 2, 562.

[9] Xu, B.; Wang, W.; Guo, L.; Chen, G.; Wang, Y.; Zhang, W.; Li, Y. Evaluation of Deep Learning for Automatic Multi-View Face Detection in Cattle. Agriculture 2021, 11, 1062.

[10] Kumar, S.; Singh, S.K.; Abidi, A.I.; Datta, D.; Sangaiah, A.K. Group Sparse Representation Approach for Recognition of Cattle on Muzzle Point Images. Int. J. Parallel Program. 2017, 46, 812–837. 35

[11] Shahinfar, S.; Meek, P.; Falzon, G. “How many images do I need?” Understanding how sample size per class affects deep learning model performance metrics for balanced designs in autonomous wildlife monitoring. Ecol. Inform. 2020, 57, 101085.

[12] Hansen, M.F.; Smith, M.L.; Smith, L.N.; Salter, M.G.; Baxter, E.M.; Farish, M.; Grieve, B. Towards on-farm pig face recognition using convolutional neural networks. Comput. Ind. 2018, 98, 145–152.

[13] Wu, Dihua, Qian Wu, Xuqiang Yin, Bo Jiang, Han Wang, Dongjian He and Huaibo Song. “Lameness detection of dairy cows based on the YOLOv3 deep learning algorithm and a relative step size characteristic vector.” Biosystems Engineering 189 (2020): 150-163.

[14] Kusakunniran, Worapan and Thanatchon Chaiviroonjaroen. “Automatic Cattle Identification based on Multi-Channel LBP on Muzzle Images.” 2018 International Conference on Sustainable Information Engineering and Technology (SIET) (2018): 1-5.

[15] Kumar, Santosh, Sanjay Kumar Singh and Ashutosh Kumar Singh. “Muzzle point pattern based techniques for individual cattle identification.” Iet Image Processing 11 (2017): 805-814.

[16] El-Henawy, Ibrahim Mahmoud, Hazem. M. El Bakry and Hagar M. El Hadad. “Cattle Identification Using Segmentation-based Fractal Texture Analysis and Artificial Neural Networks.” (2016).

[17] Kusakunniran, Worapan, Anuwat Wiratsudakul, Udom Chuachan, Sarattha Kanchanapreechakorn, Thanandon Imaromkul, Noppanut Suksriupatham and Kittikhun Thongkanchorn. “Biometric for Cattle Identification using Muzzle Patterns.” Int. J. Pattern Recognit. Artif. Intell. 34 (2020): 2056007:1-2056007:21.