VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

CSE3088

ARTIFICIAL INTELLIGENCE

PROJECT TITLE:

CROP PROTECTION FROM ANIMALS

Team Members:

Nithya Sharma - 19MIA1028
Tharani Kumar – 19MIA1033
Parvathy AJ – 19MIA1048

Faculty Name:
Dr. SL Jayalakshmi

Slot: B2

INDEX

ACKNOWLEDGMENT

Primarily, we would like to thank the almighty for all the blessings he showered over us to complete this project without any flaws.

The success and final outcome of this assignment required a lot of guidance and assistance from many people and we are extremely fortunate to have got this all along with the completion of our project. Whatever we have done is only due to such guidance and assistance by our faculty, Dr. S.L.JAYALAKSHMI, to whom we are really thankful for giving us an opportunity to do this project.

Last but not the least, we are grateful to all our fellow classmates and our friends for the suggestions and support given to us throughout the completion of our project.

ABSTRACT

Agriculture is a means of employment for 58% of India’s population, a key contributor to the country’s GDP and a source of raw materials for the country’s industries. Crop damage is a problem that leads to many losses for farmers and an efficient method to ensure complete security of the crops has not yet been developed. In this project, we have proposed a system based on Artificial Intelligence that works to protect the crops from animals. The system works on a web application with a live video stream of the farmlands. A Convolutional Neural Networks model is used to detect animals in the feed. When an animal that could potentially damage crops is discovered, an alarm sound is emitted. This helps in alerting the farmers to the intruders and thus assists in preventing crop damage in such a way that the animals are also not harmed.

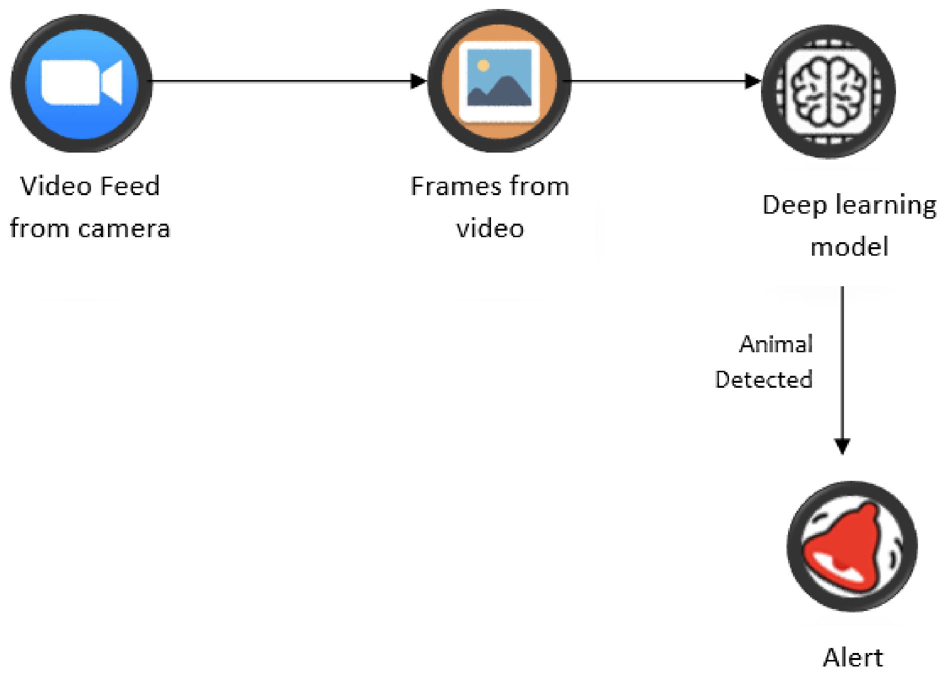
INTRODUCTION

The field of agriculture is the primary source of livelihood for more than half of India’s population and a great contributor to the economy. However, the lives of farmers are rife with many toils and troubles. One among the many problems is the damage inflicted on crops by animals. Whether the damage occurs due to wild and local animals consuming the crops, uprooting them or stampeding them, it is a significant problem which farmers find difficult to face.

The farmlands are vast and there are never-ending tasks to be done in them, so it is impossible for the farmers to constantly monitor the crops. Manual methods of protection are also difficult, since their efficiency is not the best and there is potential danger to farmers in chasing away certain ferocious animals. Therefore, it is necessary to come up with a system which can work efficiently to protect the crops.

It must also be noted that no harm should come to the animals because of the system, as several crop protection methods include electric fences and similar features which cause injury and fatality to the animals. Thus, an ethical system must be arrived at which optimizes crop protection while ensuring minimal damage to the animals.

ARCHITECTURAL DIAGRAM

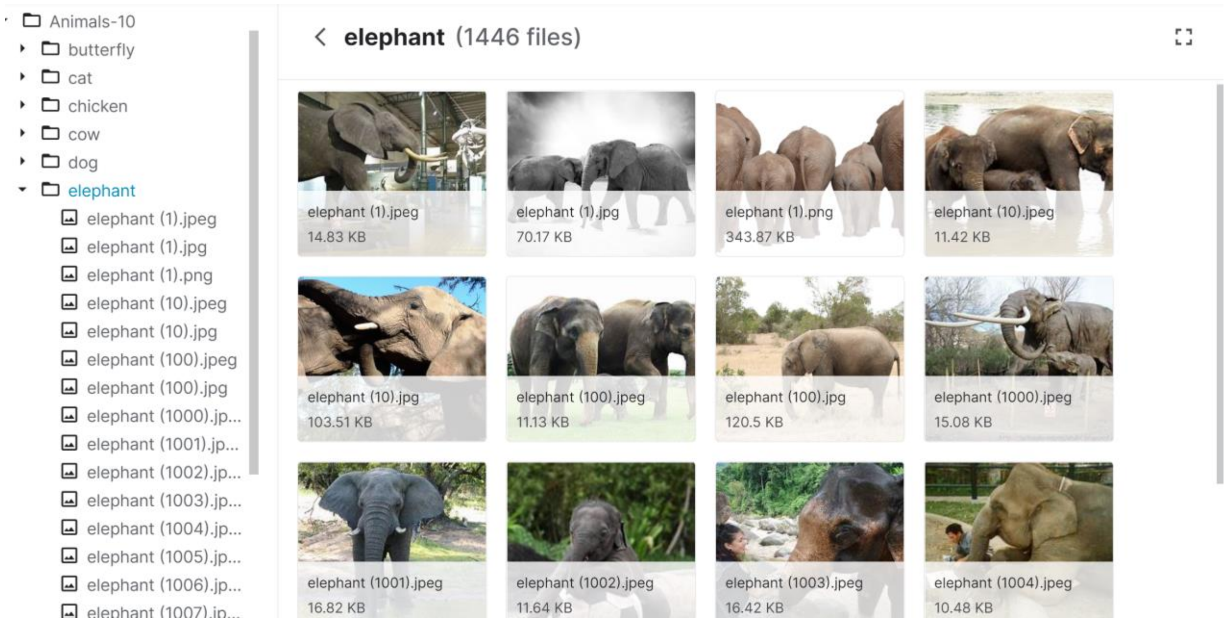


The system includes 24/7 surveillance of the crops and farmlands via cameras installed in the area. The content captured by the cameras will be set up to be live streamed to a web application.

The web application would process each frame of the incoming video stream and work to detect any animals that may be intruding on the lands. If an animal were to be detected, a loud alarm sound would be played, thus alerting the farmer to the situation and startling the approaching animals. This ensures the safety of the crops and helps in avoiding or at the very least, minimizing damage to the crops.

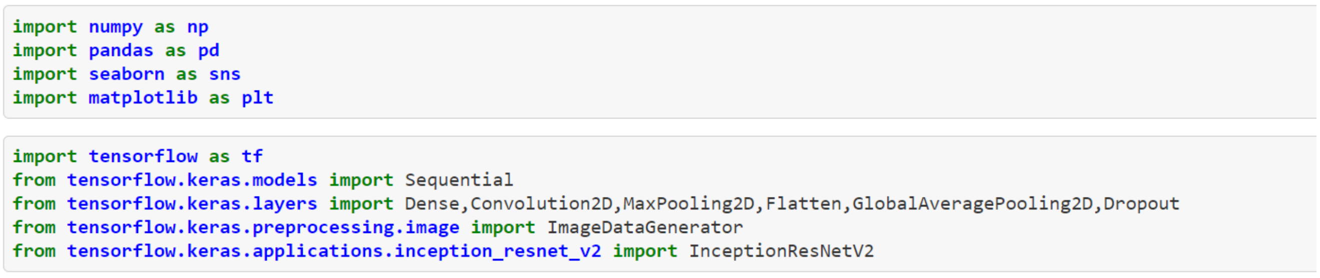
MODULES

1) Dataset Information:



The dataset used in this project was taken from Kaggle, and it is a collection of images of ten different animals, namely: Butterfly, cat, chicken, cow, dog, elephant, horse, sheep, spider and squirrel. The dataset contains over 1500 images of each category of animals, with a total of 26.2k files. The link to the dataset is as follows: https://www.kaggle.com/viratkothari/animal10

2) Importing Libraries:

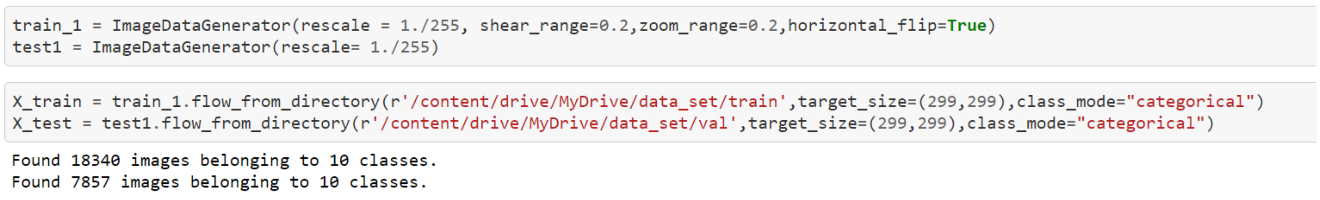


```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib as plt
```

```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Convolution2D,MaxPooling2D,Flatten,GlobalAveragePooling2D,Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.inception_resnet_v2 import InceptionResNetV2
```

The libraries required for carrying out the project are imported. Numpy and Pandas are used for data preprocessing and analysis.
Tensorflow is used for video detection, image recognition as well as classification and prediction of animals.

3) Splitting of dataset:



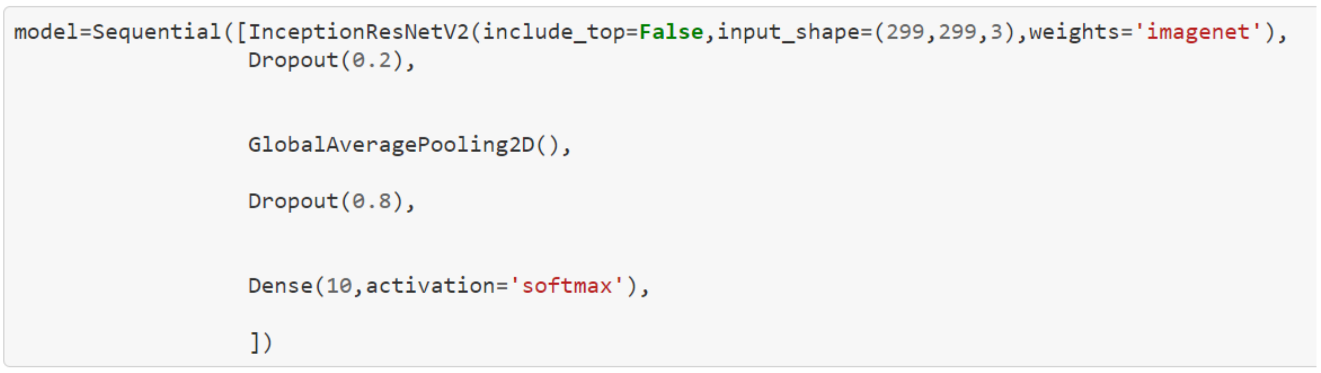
```python
train_1 = ImageDataGenerator(rescale = 1./255, shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
test1 = ImageDataGenerator(rescale= 1./255)
```

```python
X_train = train_1.flow_from_directory(r'/content/drive/MyDrive/data_set/train',target_size=(299,299),class_mode="categorical")
X_test = test1.flow_from_directory(r'/content/drive/MyDrive/data_set/val',target_size=(299,299),class_mode="categorical")
```

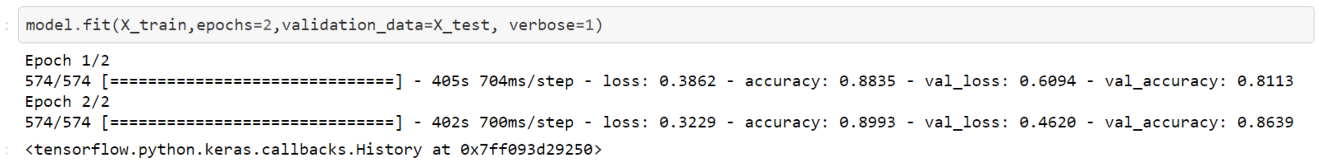
```
Found 18340 images belonging to 10 classes.
Found 7857 images belonging to 10 classes.
```

The dataset is split into train and test sets. ImageDataGenerator is used to load and augment the images for classification and further split them. Out of the 26,000+ files, 18,340 belong to the train set and 7,857 in the test.
These animals can be classified into 10 classes.

4) Training the Model:



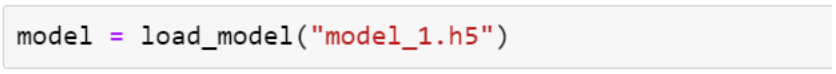
```
model=Sequential([InceptionResNetV2(include_top=False,input_shape=(299,299,3),weights='imagenet'),
                  Dropout(0.2),


                  GlobalAveragePooling2D(),

                  Dropout(0.8),


                  Dense(10,activation='softmax'),

                  ])
```



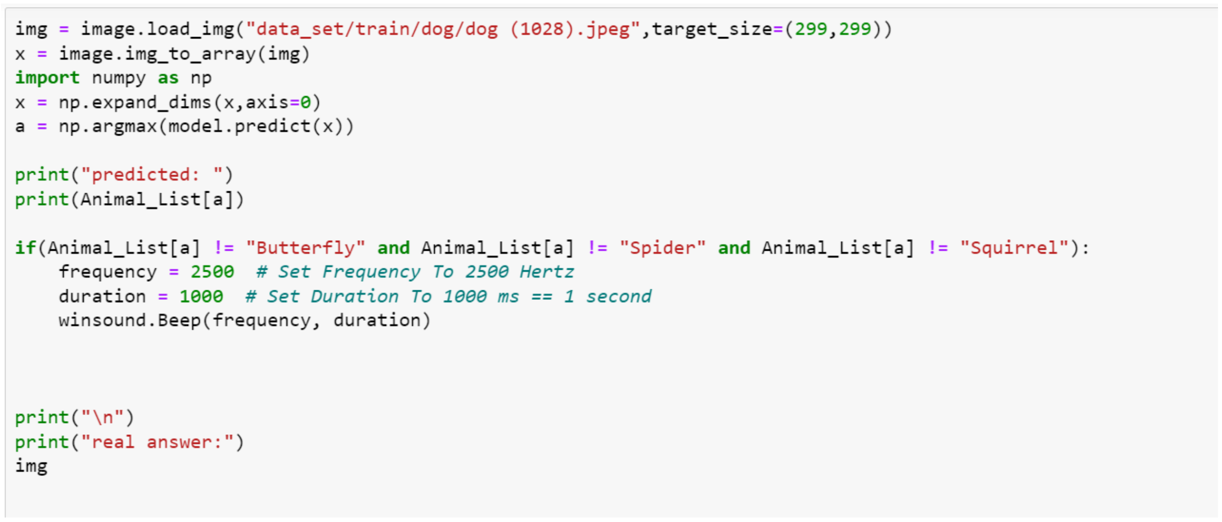
```
model.fit(X_train,epochs=2,validation_data=X_test, verbose=1)

Epoch 1/2
574/574 [==============================] - 405s 704ms/step - loss: 0.3862 - accuracy: 0.8835 - val_loss: 0.6094 - val_accuracy: 0.8113
Epoch 2/2
574/574 [==============================] - 402s 700ms/step - loss: 0.3229 - accuracy: 0.8993 - val_loss: 0.4620 - val_accuracy: 0.8639
<tensorflow.python.keras.callbacks.History at 0x7ff093d29250>
```

The training model used Transfer Learning. InceptionRedNetV2 is the pre-trained model from which the weights were taken. The input was reshaped into (299,299,3) so they are uniform. The Dropout Layer was used to make sure there is no overfitting. GlobalAveragePooling2D gives the average of all the filters that were identified from the data. Dense layer used 'Softmax' for classification of data. The number of neurons were takes as 10 (10 classes).
The model was fit and later saved into a file called 'model_1.h5'.

5) Testing the Model:



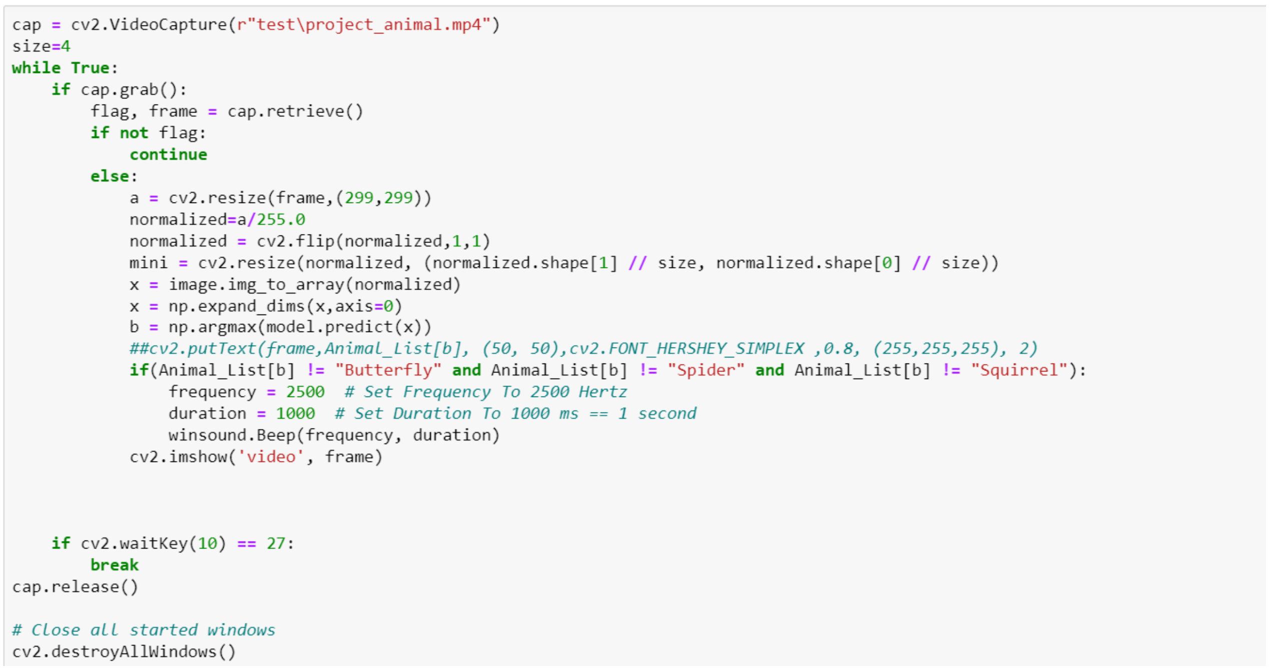
```
model = load_model("model_1.h5")
```



```
img = image.load_img("data_set/train/dog/dog (1028).jpeg",target_size=(299,299))
x = image.img_to_array(img)
import numpy as np
x = np.expand_dims(x,axis=0)
a = np.argmax(model.predict(x))

print("predicted: ")
print(Animal_List[a])

if(Animal_List[a] != "Butterfly" and Animal_List[a] != "Spider" and Animal_List[a] != "Squirrel"):
    frequency = 2500  # Set Frequency To 2500 Hertz
    duration = 1000  # Set Duration To 1000 ms == 1 second
    winsound.Beep(frequency, duration)



print("\n")
print("real answer:")
img
```

An image is loaded into the model from the dataset and was further converted into an array using numpy. The model is later made to predict the animal in the picture and if the animal is classified to be anything other than Butterfly, Spider and Squirrel, a beep sound is produced and the model predicts the animal shown in the picture.

6) Application:



```python
cap = cv2.VideoCapture(r"test\project_animal.mp4")
size=4
while True:
    if cap.grab():
        flag, frame = cap.retrieve()
        if not flag:
            continue
        else:
            a = cv2.resize(frame,(299,299))
            normalized=a/255.0
            normalized = cv2.flip(normalized,1,1)
            mini = cv2.resize(normalized, (normalized.shape[1] // size, normalized.shape[0] // size))
            x = image.img_to_array(normalized)
            x = np.expand_dims(x,axis=0)
            b = np.argmax(model.predict(x))
            ##cv2.putText(frame,Animal_List[b], (50, 50),cv2.FONT_HERSHEY_SIMPLEX ,0.8, (255,255,255), 2)
            if(Animal_List[b] != "Butterfly" and Animal_List[b] != "Spider" and Animal_List[b] != "Squirrel"):
                frequency = 2500  # Set Frequency To 2500 Hertz
                duration = 1000  # Set Duration To 1000 ms == 1 second
                winsound.Beep(frequency, duration)
            cv2.imshow('video', frame)



    if cv2.waitKey(10) == 27:
        break
cap.release()

# Close all started windows
cv2.destroyAllWindows()
```

The video is imported using the VideoCapture function and cap.retrieve is used to retrieve the frames of the video. The video is then resized to (299,299) as all the images in the dataset are resized to the same dimensions. Then the video is normalized and since normalization can cause the video to get inverted, it is flipped. If the animal detected in the frame is classified to be anything other than Butterfly, Spider or Squirrel (animals capable of harming the crop), a beep sound is produced with a frequency of 2,500 Hz which lasts for 1 second per frame.

TECHNOLOGIES AND INTEGRATION

The tools and programming languages used in this project are as follows:

1) NUMPY:

NumPy is a python module that is used for scientific computing. It has functions that can help work with linear algebra, matrices and fourier transform. In the model, this package is used to convert the data into arrays.

2) PANDAS:

Pandas is a python library that is mainly used for data analysis. It allows the user to import datasets and perform various data manipulation operations on them like merging, reshaping, etc.

3) TENSORFLOW:

Tensorflow is an opensource software library for machine learning and artificial intelligence. It is used for building models using data flow graphs. It is used for classification, perception and prediction of data. Tensorflow version 2.x is required to convert the arrays into tensors, construct and build the model, add layers and to preprocess the data.

4) OpenCV:

OpenCV is a programming function library that is used for solving computer vision problems. In the web application development process and integration with the model, OpenCV is required for video capturing and generating frames from them to run the model on.

5) HTML:

HyperText Markup Language or HTML is used to structure and code a web page. It creates e-documents that can be displayed on a webpage. In this project, HTML is used to design the webpage of the application with the help of CSS, which is a stylesheet language.

6) FLASK:

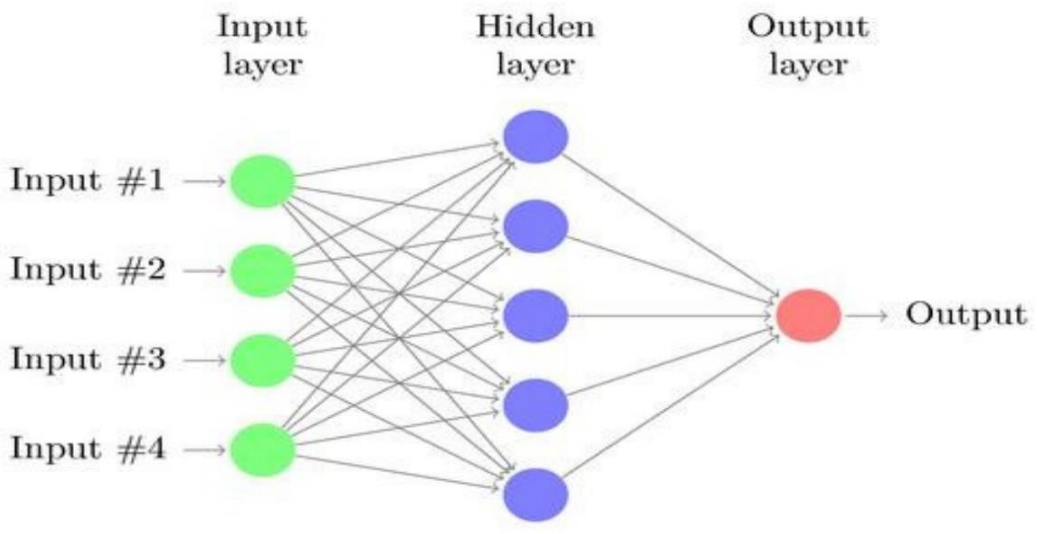
Flask is a framework that helps the user build web applications. Flask is required to implement the program on a HTML page. Flask's render template is for rendering the html page, response is for responding to html page with video feed, load_model is used to load the trained and stored model and winsound is used to produce an alert sound whenever an animal is detected.

7) JUPYTER NOTEBOOK:

It is an open-sourced web application that allows the user to share and create documents with codes, visualizations, etc. It is also used for data cleaning and transformation, statistical modeling and visualization. It was used in this project to build and save the model which was deployed in the application.

IMPLEMENTATION DETAILS

Convolutional Neural Networks (CNN):



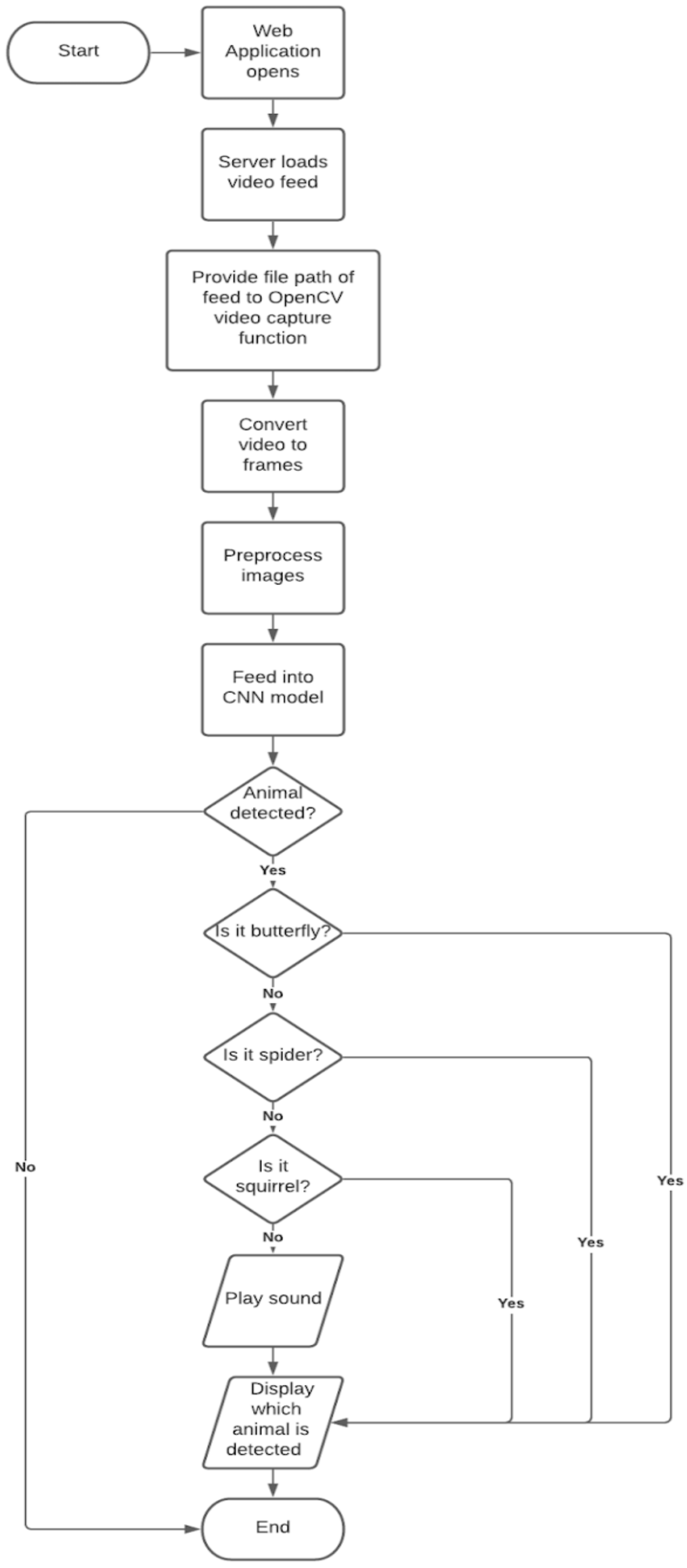
Neural networks are essentially computer programs that are modeled in a similar fashion as to how a human brain works. Like our brain, neural networks consist of a whole bunch of neurons that can’t do much individually, but when connected together in a network, can do some pretty incredible tasks. The job of a neuron is quite simple, it takes in a couple inputs; let’s call them X1, X2, and X3, X4, as per the diagram, and then spits out an output. Each of the inputs has a certain weight coefficient associated with them. These weights affect the output of the neuron and they are constantly changed to get an improved output.

CNN are different from normal neural networks because they contain a special type of layer called a Convolution Layer, which contains a filter that is able to understand certain types of patterns in the image. Usually, layers that are right at the start would be used for something pretty simple like edge detection. Later on, layers can detect more complex features like eyes, noses, and fingers. All of these feature maps are put into a long list of features at the end of the network, which is used to classify the given image.
The web application has a live feed of the farmlands in the form of a video feed built into the server. The path of the feed is captured and provided to the OpenCV video capture function. This function is used to convert the video into images (each frame of the video is converted to an image) by retrieving each frame and resizing it to 299 pixels. The image is then pre-processed, normalizing it and flipping it when necessary. The goal of this is to customize the image in such a way that the model is able to understand it and work with it efficiently.

When the image is ready, it is fed into the CNN model. The CNN model is built to scan the images and detect the presence of animals. It can detect a total of ten animals, namely, Butterfly, Cat, Chicken, Cow, Dog, Elephant, Horse, Sheep, Spider and Squirrel. If an animal is detected, it returns the index corresponding to said animal (the indices correspond to the alphabetical order of the animal).

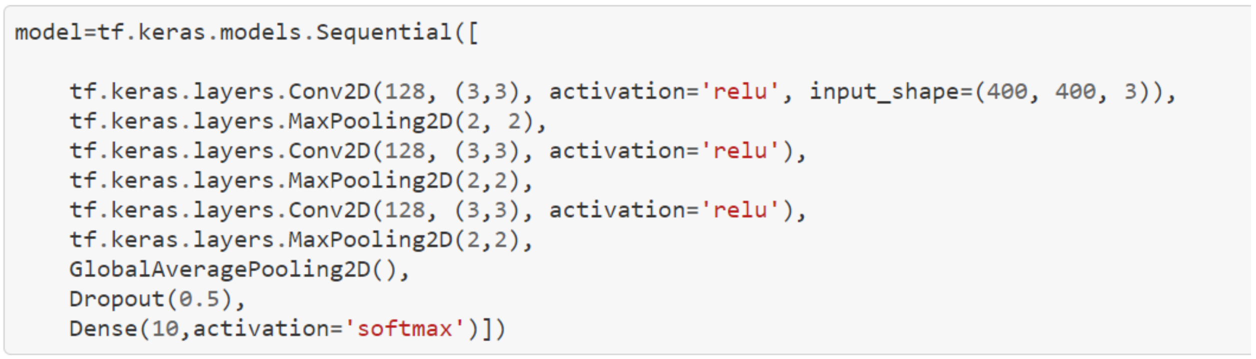
If an animal is detected in the given image by the CNN model, an alarm is sounded, unless the animal which is detected is a butterfly, spider or squirrel. The alarm is generated using ultrasonic sound waves at a frequency of 2500 Hz for a duration of 1000 milliseconds (1 second). A message showing what type of animal is detected will also be displayed.

Flow Chart:


Start
Web Application opens
Server loads video feed
Provide file path of feed to OpenCV video capture function
Convert video to frames
Preprocess images
Feed into CNN model
Animal detected?
Yes
Is it butterfly?
No
Is it spider?
No
Is it squirrel?
No
Play sound
Display which animal is detected
End
No
Yes
Yes
Yes

RESULT AND DISCUSSION

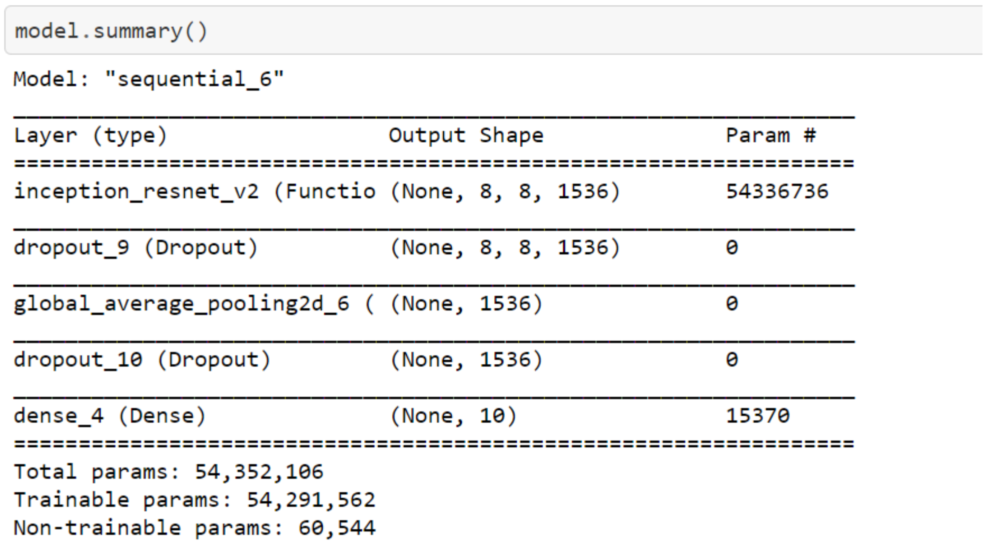
Initial Model (Not Used):



```
model=tf.keras.models.Sequential([

    tf.keras.layers.Conv2D(128, (3,3), activation='relu', input_shape=(400, 400, 3)),
    tf.keras.layers.MaxPooling2D(2, 2),
    tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    GlobalAveragePooling2D(),
    Dropout(0.5),
    Dense(10,activation='softmax')])
```

The model was built with six layers of convolution, which gave training and testing accuracies to be 67% and 76% respectively.
Since the other model gave better accuracy, we decided to the next model.

Model:

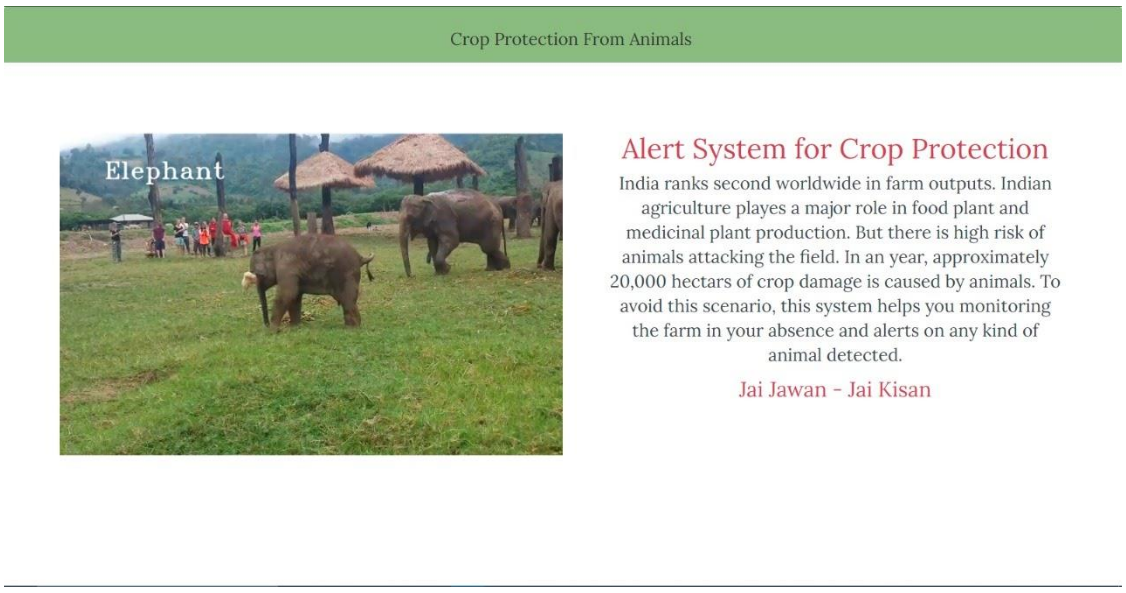


```
model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
inception_resnet_v2 (Functio (None, 8, 8, 1536)        54336736
_________________________________________________________________
dropout_9 (Dropout)          (None, 8, 8, 1536)        0
_________________________________________________________________
global_average_pooling2d_6 ( (None, 1536)              0
_________________________________________________________________
dropout_10 (Dropout)         (None, 1536)              0
_________________________________________________________________
dense_4 (Dense)              (None, 10)                15370
=================================================================
Total params: 54,352,106
Trainable params: 54,291,562
Non-trainable params: 60,544
```

A training accuracy of 88.54% and a testing accuracy of 68% were reached by using the optimizer, “Adam” with 100 epochs being used, along with batch sizes of 256 for the training set and 64 for the testing set.
When 50 epochs were used with batch sizes of 128 for the training set and 32 for the testing set, accuracies of 76% and 67% were reached for training and testing respectively.
The model achieved a training accuracy of 91% and testing accuracy of 87% after applying transfer learning on it. Inception Resnet V2 was used for transfer learning and ‘imagenet’ was used for the weights. The activation function used was ‘Softmax’. This model was chosen for the system.

Web Application:


Crop Protection From Animals

Elephant

Alert System for Crop Protection

India ranks second worldwide in farm outputs. Indian agriculture playes a major role in food plant and medicinal plant production. But there is high risk of animals attacking the field. In an year, approximately 20,000 hectars of crop damage is caused by animals. To avoid this scenario, this system helps you monitoring the farm in your absence and alerts on any kind of animal detected.

Jai Jawan - Jai Kisan

The above image shows a screen capture of the website. On the left, the live video feed of the farmlands is depicted and when an animal is detected, it displays the name of the animal on the feed, as well as below it. On the right, a brief description about the Indian agriculture feed is visible. When an animal capable of destroying crops in detected by the system, it produces a beep sound which lasts for 1 second and can hence notify farmers about the invasion.

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION

The web application is user friendly and successfully depicts the output of the system. It is simple, easy to understand and made accessible to all kinds of users. The live video feed of the farmland ensures constant and vigilant monitoring of the crops. Manual labour required by farmers or workers in patrolling the field can be greatly reduced due to this.

The model used in the project has achieved an accuracy of 87% and is able to successfully detect the presence of an animal in a given image and convey what animal has been detected to the user. The alarm sound is sufficient to alert the farmers regarding the animal intrusion.

Thus, the system is capable of efficiently protecting crops from damage by animals, while not causing harm to the animals as well. It will provide a smart solution to the woes of the farmers, especially as the government is also working to assist them through the Digital India movement.

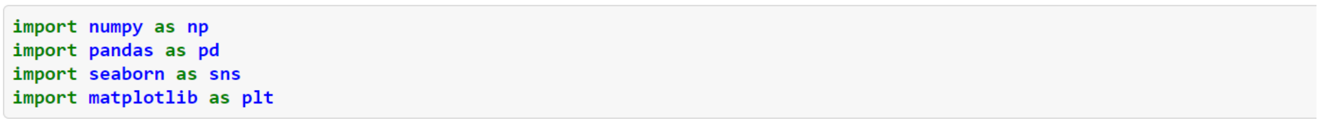
FUTURE SCOPE

In the future, practical applications of this system can be made in farmlands, with cameras and a web network supporting a live feed. The system could be integrated with the Cloud and a record could be maintained of the different animal intrusions, which can be used to predict the attacks based on factors such as time period where attacks are frequent, type of crops grown, fertilizers used, etc. The reach of what the system can detect can also be expanded to birds and pests.

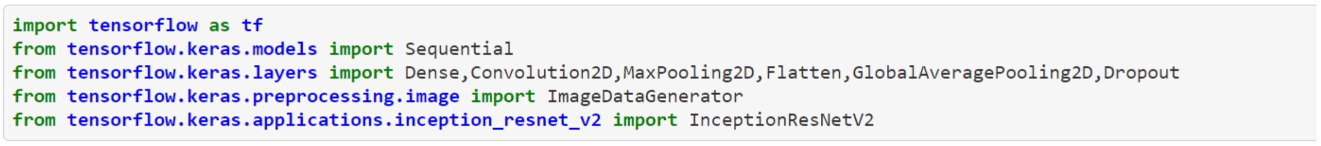
Endangered species of animals can also intrude upon farmlands and damage crops, in cases where there are wildlife sanctuaries or national parks nearby. A collection of endangered animals can be added to the system and when such an animal is detected, an alert could be sent to the forest officials nearby, who can safely guide the animals back to their habitats.

If a certain pest infestation is prominent in the field, the system can be configured to recommend a pesticide or insecticide to the farmer on the web application. The user could input the type of crops grown so that a pesticide could be recommended which does not cause harm to the crops. Economical home remedy recipes for getting rid of pests can also be suggested.

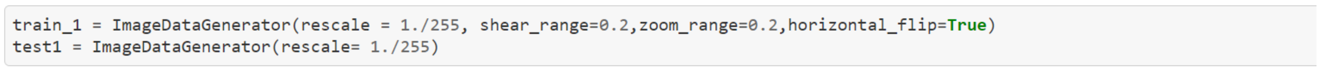
SAMPLE CODE



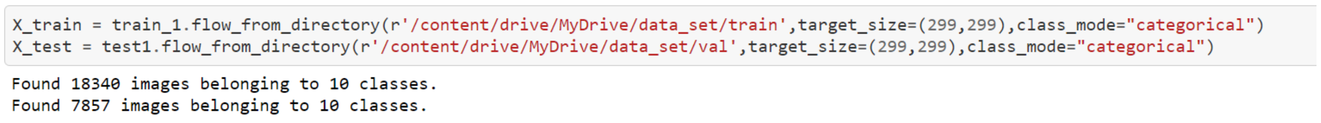
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib as plt
```



```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Convolution2D,MaxPooling2D,Flatten,GlobalAveragePooling2D,Dropout
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.inception_resnet_v2 import InceptionResNetV2
```

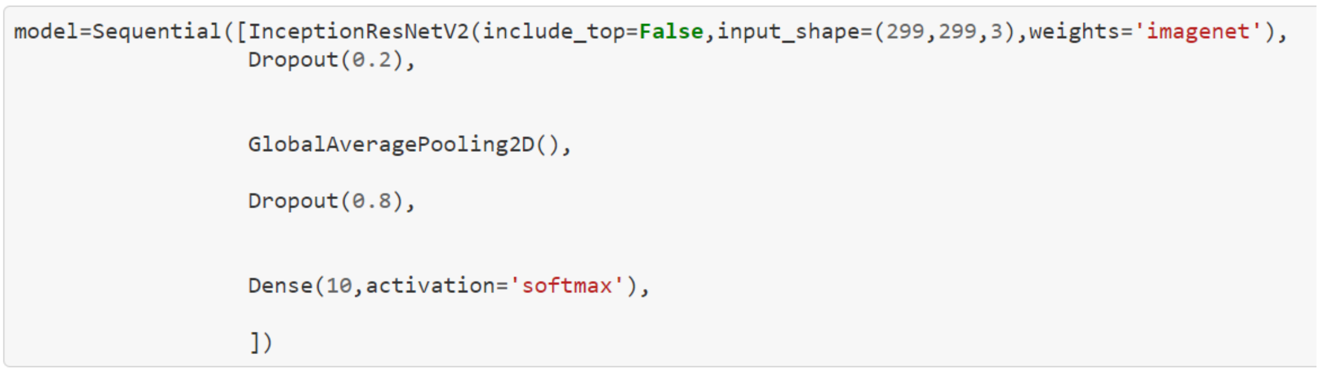


```
train_1 = ImageDataGenerator(rescale = 1./255, shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
test1 = ImageDataGenerator(rescale= 1./255)
```

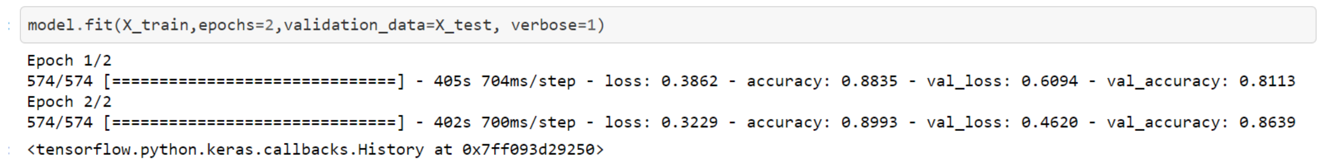


```
X_train = train_1.flow_from_directory(r'/content/drive/MyDrive/data_set/train',target_size=(299,299),class_mode="categorical")
X_test = test1.flow_from_directory(r'/content/drive/MyDrive/data_set/val',target_size=(299,299),class_mode="categorical")
```

```
Found 18340 images belonging to 10 classes.
Found 7857 images belonging to 10 classes.
```

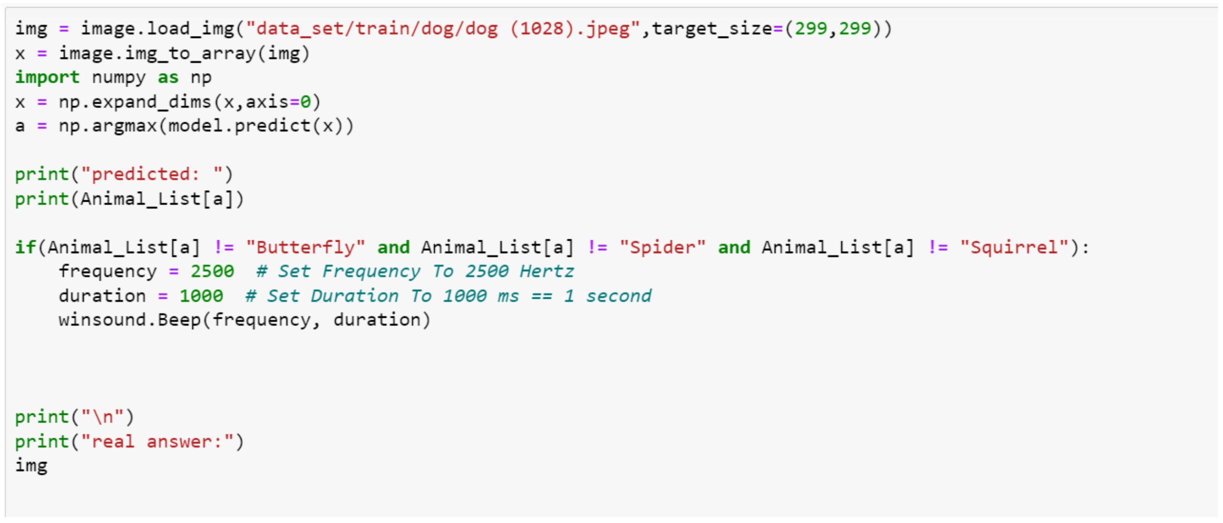


```
model=Sequential([InceptionResNetV2(include_top=False,input_shape=(299,299,3),weights='imagenet'),
                  Dropout(0.2),


                  GlobalAveragePooling2D(),

                  Dropout(0.8),


                  Dense(10,activation='softmax'),

                  ])
```



```
model.fit(X_train,epochs=2,validation_data=X_test, verbose=1)
```

```
Epoch 1/2
574/574 [==============================] - 405s 704ms/step - loss: 0.3862 - accuracy: 0.8835 - val_loss: 0.6094 - val_accuracy: 0.8113
Epoch 2/2
574/574 [==============================] - 402s 700ms/step - loss: 0.3229 - accuracy: 0.8993 - val_loss: 0.4620 - val_accuracy: 0.8639
<tensorflow.python.keras.callbacks.History at 0x7ff093d29250>
```



```
img = image.load_img("data_set/train/dog/dog (1028).jpeg",target_size=(299,299))
x = image.img_to_array(img)
import numpy as np
x = np.expand_dims(x,axis=0)
a = np.argmax(model.predict(x))

print("predicted: ")
print(Animal_List[a])

if(Animal_List[a] != "Butterfly" and Animal_List[a] != "Spider" and Animal_List[a] != "Squirrel"):
    frequency = 2500  # Set Frequency To 2500 Hertz
    duration = 1000  # Set Duration To 1000 ms == 1 second
    winsound.Beep(frequency, duration)



print("\n")
print("real answer:")
img
```

```
cap = cv2.VideoCapture(r"test\project_animal.mp4")
size=4
while True:
    if cap.grab():
        flag, frame = cap.retrieve()
        if not flag:
            continue
        else:
            a = cv2.resize(frame,(299,299))
            normalized=a/255.0
            normalized = cv2.flip(normalized,1,1)
            mini = cv2.resize(normalized, (normalized.shape[1] // size, normalized.shape[0] // size))
            x = image.img_to_array(normalized)
            x = np.expand_dims(x,axis=0)
            b = np.argmax(model.predict(x))
            ##cv2.putText(frame,Animal_List[b], (50, 50),cv2.FONT_HERSHEY_SIMPLEX ,0.8, (255,255,255), 2)
            if(Animal_List[b] != "Butterfly" and Animal_List[b] != "Spider" and Animal_List[b] != "Squirrel"):
                frequency = 2500  # Set Frequency To 2500 Hertz
                duration = 1000  # Set Duration To 1000 ms == 1 second
                winsound.Beep(frequency, duration)
            cv2.imshow('video', frame)



    if cv2.waitKey(10) == 27:
        break
cap.release()

# Close all started windows
cv2.destroyAllWindows()
```

REFERENCES

- Giordano, S. & Seitanidis, Ilias & Ojo, Mike & Adami, Davide & Vignoli, Fabio. (2018). IoT solutions for crop protection against wild animal attacks. 1-5. 10.1109/EE1.2018.8385275.

- R.S, Dr.Sabeenian & Deivanai, N & Mythili, B. (2020). Wild Animals Intrusion Detection using Deep Learning Techniques. 10.31838/ijpr/2020.12.04.164.

- Aishwarya, Kavita H M, Rashmi Reddy K, Srikanth N, Soumya D B. (2019). Smart Crop Protection System from Animals and Fire using Arduino. International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE) Vol 6, Issue 4.

- S. Vidhya, Vishwashankar, T. J., Akshaya, K., Premdas, A., and Rohith, R. (2019) Smart crop protection using deep learning approach, International Journal of Innovative Technology and Exploring Engineering, vol. 8, pp. 301-305.

- Sunil D., Arjun R., Ashokan A., Zakir F., John N.P. (2021) Smart Crop Protection System from Birds Using Deep Learning. In: Hassanien A.E., Bhattacharyya S., Chakrabati S., Bhattacharya A., Dutta S. (eds) Emerging Technologies in Data Mining and Information Security. Advances in Intelligent Systems and Computing, vol 1300. Springer, Singapore.

- Abhishek Paliwal, Omkar Kekre, Rohit Hazare, Srushti Mehatkar, (2020) International Research Journal of Engineering and Technology (IRJET) Volume: 07 Issue: 05.

- K.S. Rao, R.K. Maikhuri, S. Nautiyal, K.G. Saxena. (2002) Crop damage and livestock depredation by wildlife: a case study from Nanda Devi Biosphere Reserve, India, Journal of Environmental Management, Volume 66, Issue 3.

- Piyush Mehta,Negi Arun, Chaudhary Rashmi, Janjua Yasmin and Thakur Pankaj. (2008) “A Study on Managing CropDamage by Wild Animals in Himachal Pradesh”, International Journal of Agriculture Sciences. Volume 10, Issue 12.