WILD ANIMAL DETECTION IN FARMLANDS USING SVM

[1]Abikrishnan M. P., [2]Karthik [3]Jayanthan K. S. [4]Harikrishnan C. J
[5]Unnikrishnan E. H.
[1,2,3,4,5]Department of Computer Science and Engineering
Sreepathy Institute of Management & Technology, Vavanoor, Kerala, India
[1]abikrishnan007@gmail.com, [2]karthikkarthiz333@gmail.com, [3]jayanthan.ks@simat.ac.in,
[4]harikrishnan5718@gmail.com, [5]ukeh.hari@gmail.com

Abstract
Implementation of animal detection in farming can be used to prevent the wild animals from entering the farm fields, thereby reducing the damage caused by them to the property and the people working there. This would be of great relief to farmers since as per the reports by the All India Kisan Sabha, the loss due to these damages is considerably high and the attacks are increasing in number every year. Animal detection can be implemented using computer vision and machine learning. It will help in detecting and identifying the animal which tries to enter the field. The details which are collected in turn are sent to field owners and other officials through an android app. They could make use of these details to take necessary actions to prevent the damage that could be caused by these animals which otherwise would be of great difficulty.
Index Terms: Contour, Dataset, Support vectors, Instance, Classification, Feature.

I. INTRODUCTION

Computer vision can be applied to many fields including the medical field, remote sensing, machine vision, content based image retrieval. Computer vision tries to solve many problems in different disciplines. Computer vision is also applied in the security field to perform automatic surveillance and access control and attendance management. Computer vision can be applied in the agriculture field like disease detection of a tree by examining its leaves or flowers or fruits. Computer vision techniques can be applied in order to provide security from wild animals in agriculture. Agriculture fields near to forest areas have a severe threat from wild animals, which attacks regularly on farms. These attacks causing huge damage to agricultural crops subsequently cause significant financial losses to farmers. According to a report prepared by Himachal Kisan Sabha reported that wild animals, namely wild boars, monkeys, nilgai, and etc, were causing a loss of RS. 4500-500 crore every year to farmers in Himachal Pradesh state, India. Some measures are taken by the farmers by installing electrical fences to the farms, big floodlights in the farm and other techniques by the local farmers. Installing an electrical fence is much costlier for huge farms and kills so many animals, which is offensive legally and affects biodiversity. Other existing techniques are there but are not effective due to several reasons.

In this paper, we proposed a computer vision based solution for agriculture security from wild animals. Detecting and tracking wild animals near farm fields is done with the help of camera placed in fields and the information is sent respective authorities including the forest department. And this helps the respective authorities to take immediate actions to protect farm fields from wild animals. Humans have their eyes and their brains to see and visually sense the world around them. Computer vision is the science that aims to give similar or even better capability to machine or computer.

The remainder of this paper is organized as follows. Section II is the related works

associated with this project, Section III contains our proposed system architecture, Section IV describes the dataset we used here, Section V describes the results we obtained and Section VI concludes this paper.

II. RELATED WORKS

Machine learning is the science that make computers to act without being explicitly programmed. Machine learning algorithms are used to solve machine learning problems. Our project is also a machine learning task and there are many algorithms to solve it. In machine learning task there is no best algorithm that suits for every problem. We have to choose the best one that fit for our application by analyze their performance and our problem domain.

In our project we need a classifier to label unknown instances. In order to select the best one that give us the best result we compare different classification algorithms namely ID3, C4.5, Naive Bayes and SVM. The analysis lead us to take SVM as a best choice. Feature extraction is an important task in every learning problems. In order to do learning we need relevant data. In machine learning we represent these data as feature vectors. Our project is based on computer vision and therefore we have to convert images to corresponding feature vectors . For computer vision there are a lot of feature detection algorithms namely Harris corner detection, SIFT and SURF. We choose SIFT because its descriptors are more accurate than any other descriptors. We choose SVM and SIFT because these are fit with each other very well. SVM requires training and testing data but for our project there were no relevant dataset available. So we created our own dataset by taking images of target animals and extract their features with SIFT. We developed an android application for alerting farmers about the wild animal entry.

III. PROPOSED SYSTEM ARCHITECTURE

Proposed system contains mainly four modules. The first module deals with object detection, second module for feature extraction, third module is the SVM (Support Vector Machine) classifier and the final module is an android application by which the users get alerted about the intruder. A camera is made in focus to the field and background subtraction algorithm is used for detecting animal entry in the field. The output of this module is an image which is then passed to second module in order to extract relevant feature information that it contain. For this purpose in our project we are using SIFT (Scale-Invariant Feature Transform) algorithm. The extracted feature vector is then passed to SVM for predicting the class. Once the prediction is available, the recipients will get notified via android app.

A. Object Detection

The Background subtraction is a major preprocessing step in many vision-based applications. In our project a camera is placed in focus to the farm fields. The initial frame of the farm field is captured with the camera and stored in to the computer memory. The initial frame and all the later frames are converted to grayscale image and the Gaussian blur method is applied for the smoothing of the images. Absolute difference between current frame and the initial frame are computed and finding out the contour area of the detected objects. Obtained absolute difference is greater than a threshold value it indicating a high probability of animal in that area. The Features are extracted from this frame and then passed to SVM for classification.

RGB to GRAY : Y = 0:299R+0:587G+0:114B (1)

Gaussian blur method describes blurring an image by a Gaussian function. It is an effect widely used in graphics software, mainly to reduce the image noise and to reduce detail. In Gaussian Blur operation, the image is convolved with a Gaussian filter instead of using the box filter. The Gaussian filter is a low-pass filter that removes the high-frequency components from the image [1].

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \qquad (2)$$

- G is the Gaussian Blur operator
- x,y are the location coordinates
- σ is the ”scale” parameter, the amount of blur. If the value increases, blur will be increased

Contour detection in OpenCV provides a function called FindContours which will helps to find the contours in the images. Contours can be simply explained as a curve that joining all the

continuous points (along the boundary), that having same color or intensity. The contours are useful tool for shape analysis and object detection and recognition [7].

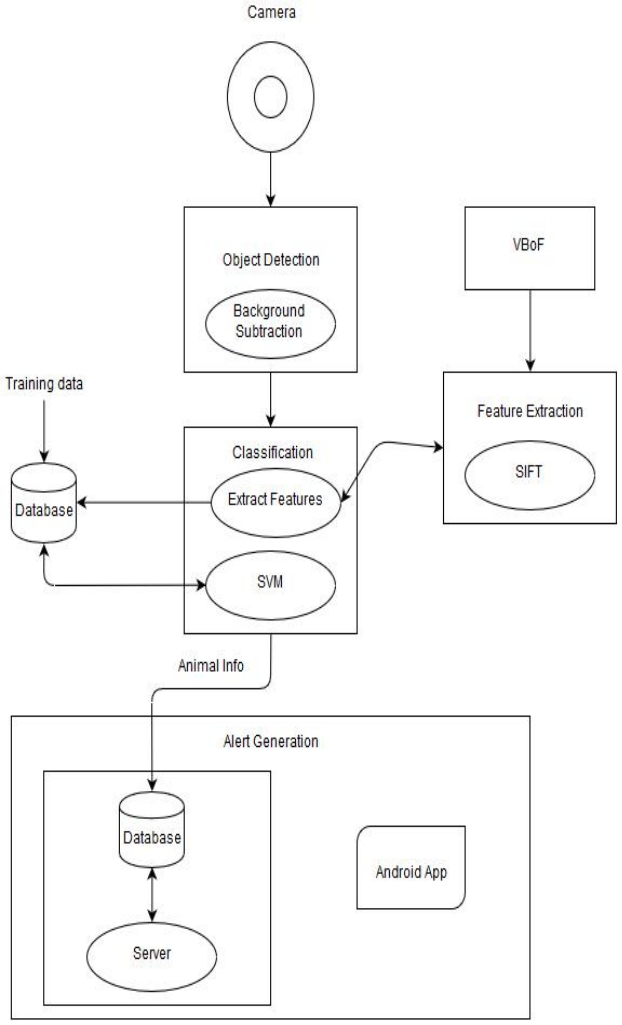

Fig. 1. System Architecture

B. Feature Extraction

In simple words features are the unique properties that defines an image or unique signature of a given image. Features are extracted from an image in order to differentiate between the images [5]. Feature extraction and matching is at the bottom of the many PC vision issues, such as seeing and stereo matching. In our project we uses OpenCV's scale invariant feature transform (SIFT) for real-time extraction of image features. Among various feature detector, the scale invariant feature transform (SIFT) algorithm is the best approach. This algorithm is mainly applicable for multi scale images. The options extracted exploitation SIFT algorithm unit of measurement invariant to image scaling, rotation, transition and partly invariant to illumination and 3D camera read purpose. The SIFT algorithm is divided into two separate modules. They are key purpose detection module and descriptor generation module [3]. In OpenCV the window rotation technique is used. It becomes the bottleneck for key purpose detection module. In the proposed method, the novel window is split into sub regions in sixteen directions and histogram rearrangement technique is employed in descriptor generation. SIFT isn't just scale invariant. We can use another from the following and still get good results:

- Scale
- Rotation
- Illumination
- Viewpoint

1) The Scale Space: To create a scale space, we take the original image and generate progressively blurred out images [5]. Then, resize original image to half of its size and generate the blurred out images again. Repeat above process more than once.

Blurring : Mathematically, "blurring" is referred to as the convolution of the gaussian operator and the image. Gaussian blur has a particular operator or expression that is applied to each pixel [3]. What results is the blurred images

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \qquad (3)$$

- L is a blurred image
- G is the Gaussian Blur operator I is an image
- x,y are the location coordinates
- σ is the "scale" parameter. Think of it as the quantity of blur. Greater the value, greater the blur.

2. Log Approximation : In the last step, we created the scale space of the image. The idea was to blur the image progressively, shrink it, blur the small image progressively and so on. Now we use those blurred images to generate another set of images, the Difference of Gaussians (DoG). These DoG images are great for finding out interesting key points in the image.

Laplacian of Gaussian: The laplacian of Gaussian (termed as LoG) operetion goes like this. we take an image, and blur it a little. Then we calculate second order derivatives on it (or,the ”laplacian). It will identifiy edges and corners on the image. These edges and corners are good for finding keypoints.

The con: To generate Laplacian of Gaussian image quickly, we use the scale space. We calculate the difference between two consecutive scales. Or, the difference of Gaussians. Just the Laplacian of Gaussian Images aren't great. They are not scale invarient. That is,they depend on the amount of blur we do. This is because of the Gaussian expression in equation (1).

Here σ^2 in the denominator is the scale. In order to get rid of it represent the laplacian of a gaussian as follows

$$\Delta^2 G$$

Then the scale invariant laplacian of gaussian is given by

$$\sigma^2 \Delta^2 G$$

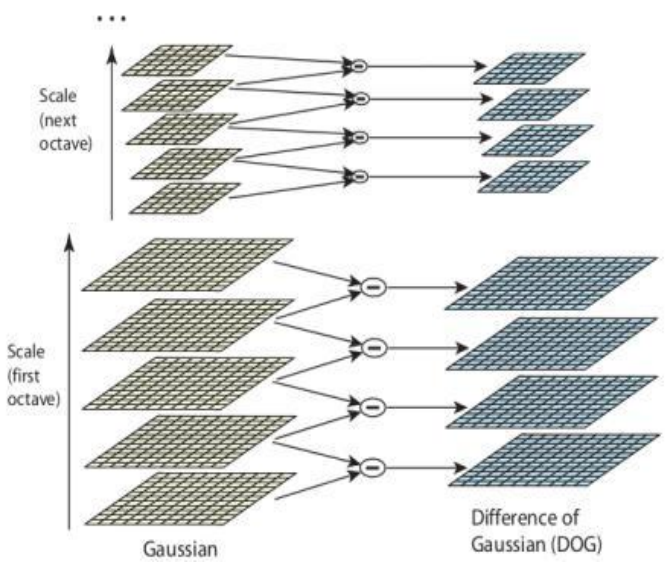

Fig. 2. DoG [5].

3. Finding keypoints: Finding key points is a two part process
1.Locate maxima/minima in DoG images
2.Find subpixel maxima/minima

Locate maxima/minima in DoG images:
Coarsely locate the maxima and minima is the first step. This is easy to understood. We iterate through each pixel and check all it's neighbours. The check is done within the current image, and also the one below and above it.

x marks the current pixel. The green circles mark the neighbours [5]. This way, a total of 26 checks are made. If it is the greatest or least of all 26 neighbours then x is marked as a ”key point”. usually, a non-maxima or non-maxima position won't have to go through all 26 checks. A few initial checks will usually sufficient to discard it. Note that key points are not detected in the topmost and lowermost scales. There simply are not enough neighbors to do the comparisons. So simply skip them. Once this is done, the marked points are the approximate minima and maxima. They are approximate because the min-ima/maxima almost never lies exactly on the same pixel [4]. It lies some where between the pixel. But we cannot access date between pixels. So, we must locate the subpixel location mathematically.

Find subpixel maxima/minima

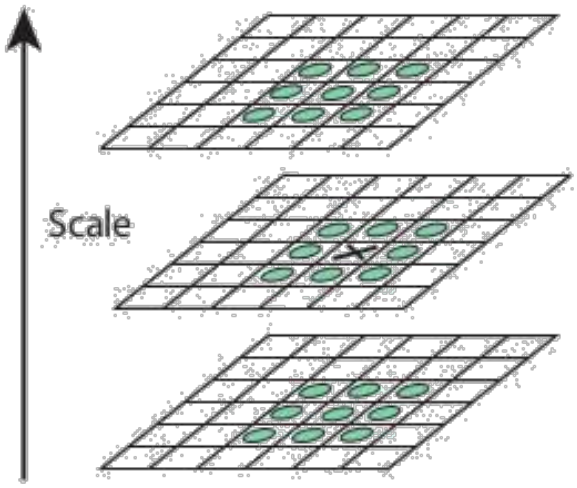

Fig. 3. Finding key points [5].

Subpixel values are generated using the available pixel data. This is done by the Taylor expansion of the image around the approximate key points. Mathematically, this can be writern as

$$D(x) = D + \frac{\gamma}{\delta x} x + \frac{1}{2} x^T \frac{\delta^2 D}{\delta x^2} \qquad (4)$$

4) Getting rid of low contrast keypoints : Key points generated in the previous step will

produce a lot of key points. Some of them don't have enough contrast, or they lie along an edge. They are not useful as features in both cases. So we get rid of them.

Removing low contrast features:

This is very simple to implement. If the intensity's magnitude (i.e., without sign) at the current pixel in the DoG image (that is being checked for minima/maxima) is less than that of a certain value, it is rejected [5].

5) Keypoint orientations : After step 4, we have legitimate key points. They've been checked to be stable. We know the scale at which the keypoint was detected (it's the same as the scale of the blurred image). So we have scale invariant with us [4]. The next thing is to assign an orientation to each and every keypoint. This orientation allows rotation invariance. The more invariance we have the better it is.

Gradient magnitudes and orientations are calculated using these formulae:

$$(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x-y,1))^2} \quad (5)$$

$$\theta(x,y) = tan^{-1}((L(x,y+1) - L(x,y+1)/L(x+1,y) - L(x-1,y))) \quad (6)$$

6) Generate sift features : Now for the final step of sift. Till now, we had scale and rotation invarience. Now we have to create a fingerprint for each and every keypoint. This is to identify a certain keypoint. If an eye is a keypoint, then using this fingerprint, we will be able to distinguish it from other set of keypoints, like noses, ears, fingers, etc.

We have to create a very unique fingerprint for the keypoint. It should be very easy to calculate. We also want it to be relatively lenient when it is being compared against other keypoints of the image. Things are never exactly same when we compare two different images [5].

To do this, a 16x16 window around the keypoint is needed. This 16x16 window is then broken into 16 4x4 windows.

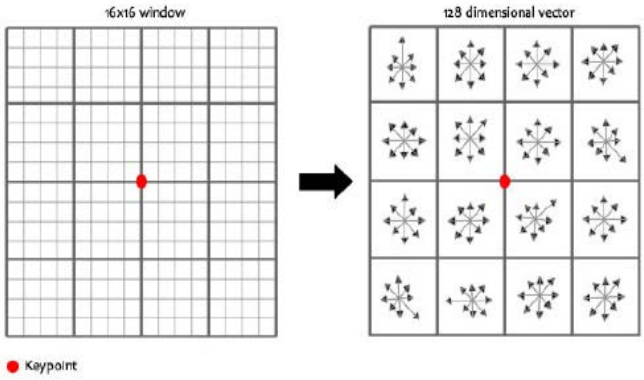

Fig. 4. Matrix representation [5].

Within each 4x4 window, gradient magnitudes and orientations are calculated. These orientations are put into an 8 bin histogram.

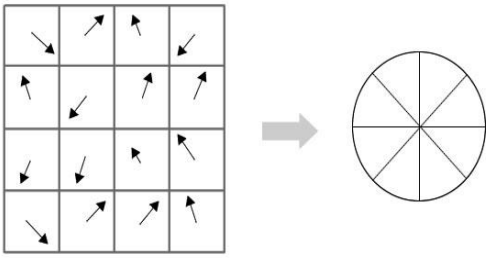

Fig. 5. sift4*4 matrix [5].

Orientation in the range 0-44 degrees add to the very first bin. 45-89 add to the second bin and so on. And (as always) the amount added to the bin depends on the gradient's magnitude. Unlike the past, the amount added is also depends on the distance from the keypoint. So gradients that are far away from the keypoint will add smaller values to the histogram compared to others [5]. This is done with the help of a "gaussian weighting function". This function will simply creates a gradient (it's like a 2-D bell curve). we multiple it with the magnitude of orientations, and we get a weighted thingy. The farther away, the lesser the magnitude.

In this project what we are doing is just extracting the features from images. The extracted keypoints(feature points) from two images are undergo comparison. The best matches will be taken out as per lowe's ratio test [8]. If these matches goes above a particular

threshold, We can say that there is some relation between these two images. In our project a testing image and BOVW image is undergo comparison. Matches count is higher than threshold indicating a great chance that that BOVW to be a part of that image.

C. Classification using SVM

A SVM is classification method based on the family of supervised learning methods. There are mainly two data types used in SVM classifier. To create a classification model training data is used. To test and evaluate the trained model accuracy, testing data are used. The main SVM classifier task is to separate training data in the higher dimensional space using a kernel function and to find an optimal hyperplane with a maximum margin between data from two different classes.

We are using LIBSVM with multiclass feature of OpenCV. LIBSVM is a popular open source library for Support Vector Machines(SVM). It was developed in the early 2000. A typical use of LIBSVM involves two steps: first, training a data set to obtain a model and second, using the model to predict information of a testing data set. This trained and tested model can then be used for classifying new entities. LIBSVM supports various SVM formulations for classification. We are using C-Support Vector Classifaction for this project.

1) C-Support Vector Classification Given training vectors $x_i \in R^n$; i = 1,……,l in two classes, and an indicator vector $y \in R^l$ such that $y_i \in \{1,-1\}$, C-SVC solves the following primal optimization problem.

$$\min_{w,b,\xi} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \tag{7}$$

$$\text{Subject to} \quad y_i \left(w^T \phi(x_i) + b \right) \geq 1 - \xi_i$$
$$\xi_i \geq 0, i = 1, \ldots, l$$

Where $\phi(x_i)$ maps xiinto a higher-dimensional space and C > 0 is the regularization parameter. Due to the possible high dimensionality of the vector variable w, usually we solve the following dual problem.

$$\min_{\alpha} \quad \frac{1}{2} \alpha^T Q \alpha - e^T \alpha \tag{8}$$

$$\text{subject to } y^T \alpha = 0$$
$$0 \geq \alpha_i \leq C, \quad i = 1, \ldots, l$$

where $e = [1; : : : ; 1]^T$ is the vector of all ones, Q is an l by l positive semidefinite matrix, Q_{ij} $y_i y_j K(x_i; x_j)$, and $K(x_i; x_j)$ $(xi)^T$ (x_j) is the kernel function. After this problem is solved, using the primal-dual relationship, the optimal w satisfies

$$w = \sum_{i=1}^{l} y_i \alpha_i \phi(x_i) \tag{9}$$

and the decision function is

$$sgn\left(w^T \phi(x) + b\right) = sgn\left(\sum_{i=1}^{l} y_i \alpha_i K(x_i, x) + b\right) \tag{10}$$

We store $y_i \alpha_i \quad \forall i, b,$ label names, support vectors, and other information such as kernel parameters in the model for prediction [6].

2) Multi-Class Classification

LIBSVM implements the one-against-one approach for multiclass classification. If k is the number of classes, then k(k 1)=2 classifiers are constructed and each one trains data from two classes. For training data from the ith and the jth classes, we solve the following two-class classification problem.

$$\min_{w^{ij}, b^{ij}, \xi^{ij}} \frac{1}{2} \left(w^{ij}\right)^T w^{ij} + C \sum_t \left(\xi^{ij}\right)_t \tag{11}$$

$$\text{subject to} \quad \left(w^{ij}\right)^T \phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij}, \text{ if } x_t \text{ in the } i\text{th class}, \left(w^{ij}\right)^T \phi(x_t) + b^{ij} \leq -1 + \xi_t^{ij}, \text{ if } x_t \text{ in the } j\text{th class}$$
$$\xi_t^{ij} \geq 0$$

In classification it uses a voting strategy: each binary classification is considered to be a voting where votes can be cast for all data points x - in the end a point is designated to be in a class with the maximum number of votes. In case that two classes have identical votes, though it may not be a good strategy, now we simply choose the class appearing first in the array of storing class names.

Multiclass SVM aims to assign labels to instances by using support vector machines, where the labels are drawn from a finite set of

several elements. The implemented approach for doing so is to reduce the single multiclass problem into multiple binary classification problems via one-versus-one.

We are using linear kernel. Linear SVM is the newest extremely fast machine learning (data mining) algorithm for solving multiclass classification problems from ultra large data sets that implements an original proprietary version of a cutting plane algorithm for designing a linear support vector machine.

The steps involved in the classification process are

1) Training: Training phase involves training the SVM using the dataset. Data set is created by comparing the set of images with the visul bag of words using SIFT [1]. This dataset is then given to SVM for training and the trained model details is stored in pickle format. As we are dealing with linear classification we are using linear kernel and hence, classes can be separated using a single Line. Training an SVM with linear kernel is faster than with any other kernel. It is one of the most common kernels to be used. It is mostly used when there are a Large number of Features in a particular Data Set. When training a SVM with a Linear Kernel, only the optimisation of the C parameter is required. C is a regularization parameter that controls the trade off between the achieving a low training error and a low testing error that is the ability to generalize our classifier to unseen data. We are taking the value of C parameter as 1.

2) Testing: Testing is used to evaluate the trained model's accuracy. we are using 20% of data set as test data and 80 % as training data. For testing unlabelled instances are passed to the trained model which labels the instances.

3) Classification: Classification phase involves classifying the new feature vector obtained using the SIFT in to most suitable class. The feature vector obtained using SIFT is given as input to the trained model [2].

D. Alerting Society and Officials: Android App Once the prediction is available from SVM, the next step is alerting the users. Our project targeting mainly two types of users. The first category is farmers and the second, the forest department. Android app is designed in such a way that the farmers can view any changes to the server database. Only wild animal entries are shared with the forest department. Admin for this app can register new users.

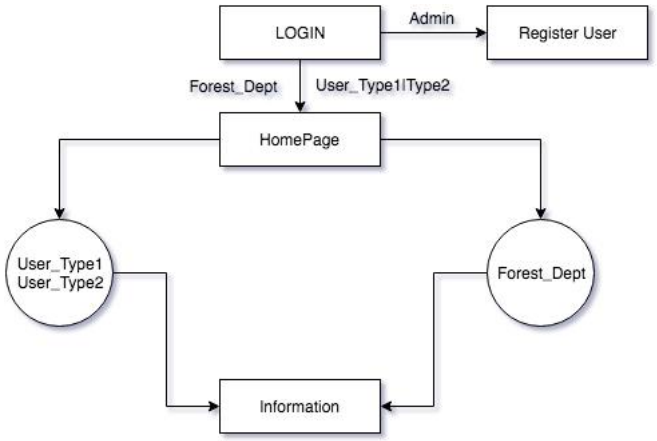

Fig. 6. Android

At the server side of our project, we use the Flask framework. Flask is a micro web framework written in Python. At the server side we defined functions for making the app-server communication possible. Our database contains three tables. The table login for storing user login details, information table for storing detected animal details and tips for storing the actions that should be taken by the users on wild animal detection. User Type1 is the primary user who is the owner of a particular camera. User Type2 is the secondary user who is connected to the User Type1's camera indirectly. Forest Dep refers to the forest department they are also a user of our android app. They are get alerted when a valid wild animal entry is detected. This alert message alsoincludesthe owneraddress of the corresponding camera. The tips table information of the animal whichis detectedis available to primary and secondary users.

IV. DATASET

Image classification can be done based on a concept called Bag of Visual Words (BOWS). Its concept was derived from information retrieval and NLP's bag of words (BOW). In bag of words (BOW), we make a count on the number of each word appears in a document, use the frequency of each word to extract the keywords from the document, and we make a frequency histogram from it. We would treat a document as a BOW. We adopt similar concept in bag of visual words

(BOVW), but instead of words, here we can use image ”features” as the words. These features are the unique pattern that we extract from an image.

The general idea is to represent an image as a set of features called bag of visual words (BOVW). Features have keypoints and descriptors. Keypoints can be viewed as the stand out points in an image. Keypoints are invariant to rotation and scale, so no matter the image is rotated, shrink, or expand, there will not any change in a keypoint. A descriptor is the description of the keypoint. We make use of these keypoints and descriptors to build vocabularies and represent each image as a frequency histogram, later we can find similarity with other images or predict the category of the image [1].

We create DATASET by comparing each training image with BOVW. For this purpose we make use of SIFT algorithm. Here we choose attributes as the animal parts that can be very useful for their detection. Attributes take binary values 0 and 1. Value 1 represents feature presence and 0 for absence.

	eear	eface	etusk	etrunk	eleg	cleg	cface	cear	gface	gleg	gear	gtail	label
2	1	1	1	1	0	1	0	1	0	0	0	0	elephant
3	1	0	0	1	1	0	0	1	0	1	0	0	elephant
4	0	0	0	0	1	1	0	1	0	1	0	1	cow
5	0	0	0	0	0	1	1	1	1	0	0	0	cow
6	0	0	1	0	0	1	0	0	1	1	0	0	goat
7	0	0	0	0	0	0	0	0	1	1	1	1	goat
8	1	0	0	0	0	0	1	0	0	0	1	0	elephant
9	1	0	1	1	0	0	0	0	1	0	1	0	elephant
10	0	0	0	0	1	0	0	0	1	1	1	0	goat
11	0	0	0	0	0	0	0	1	0	1	1	0	goat
12	0	1	0	0	0	1	1	1	0	1	0	0	cow
13	1	1	1	1	1	0	0	0	0	0	0	0	elephant

Fig. 7. Dataset

Fig. 8. BOVW

V. RESULTS AND OBSERVATIONS

Each module is tested individually and after integrating them the system test is performed for identifying errors or bugs in the system when considered as a whole. In the beginning, we turn on the camera and made it focusing on particular farmland. Set up the initial frame and compare it with the remaining for detecting object presence.

The camera helps to take images and background subtraction help to identify object region. The concept of contour is adapted to select the detected object area from the frame. The threshold applied here helps us to remove small sized objects or animals which does not cause serious damages to crops. A rectangle is drawn through the boundaries of the object area and saves it as an image. This image is then passed to SVM for getting labeled. The feature vector of the image is extracted by calling the SIFt algorithm (Feature Extraction module). The feature vector is an N attribute vector, each attribute here represents parts of each animal in our searching domain. The SVM labels the instance and put the details into the server database. When the user turns on his internet connection in his android phone, new notifications from the server are shown. On clicking, it will open the android application and displays the latest information at the top of the home screen.

The following screenshots represents results we obtained when an elephant enters to a particular farm field. SVM labeled the instance as elephant and the user’s are get informed about this through our android app.

Fig. 9. Object Detection

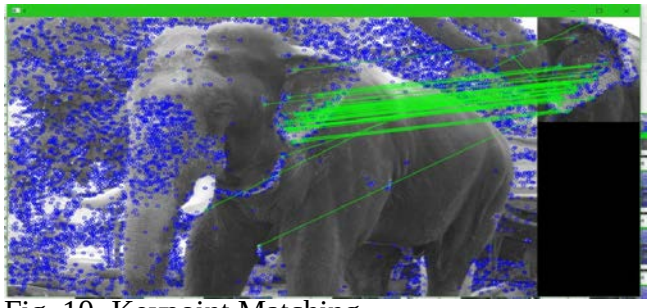

Fig. 10. Keypoint Matching

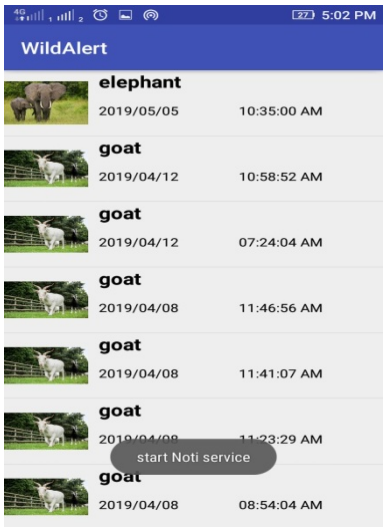

Fig. 11. Home Screen Android

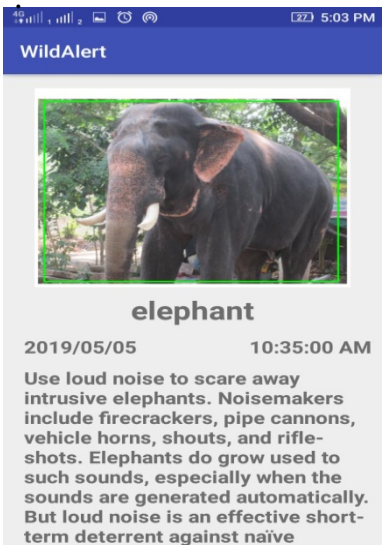

Fig. 12. Info Screen Android

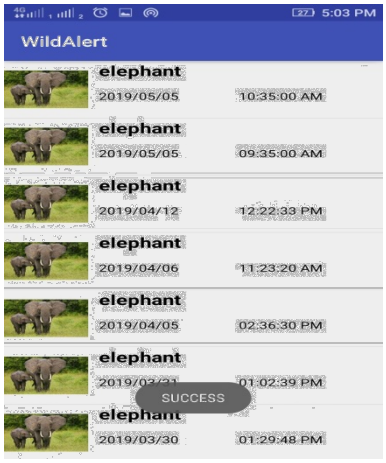

Fig. 13. Home Screen Forest Dept

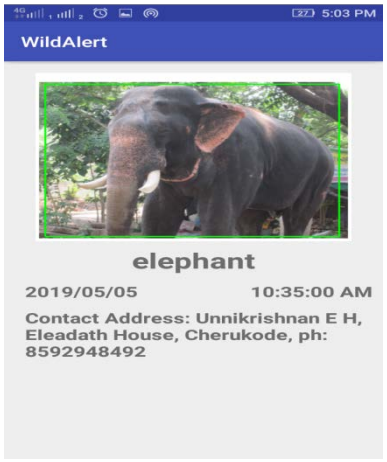

Fig. 14. Info Screen Forest Dept

VI. CONCLUSIONS

The wild animal attack in farm fields has been one of the main topics of discussion nowadays. Even though there are many methods that are currently in use to prevent this including installing an electrical fence, big floodlights, etc, but none of them are effective in considering the cost of installation or serving the purpose. Our idea uses the modern technology available today and focuses on causing no harm to the animal.

Our idea uses computer vision and machine learning techniques to identify the animal which tries to enter the farm field and we are also giving suggestion to get rid of the possible animal attack. Our tie-up with the forest department also helps to make them aware of these attacks, so that they could take measures to prevent possible attacks in the future.

Our system can be improved further by adding new functionality like adding an alarm that will beep automatically through a speaker which is installed near the camera depending on the animal identified. Different alarms can be used to make the animal stay away from the area. We can use bee sound to ran away elephants, similarly, there are other sounds that force the animal to leave the fields.

We can also enable night vision and wide area coverage by using IR 360 camera instead of a normal digital camera. With a slight modification in the algorithm, we can use the

same setup for other applications including taking census of animals in forests.

REFERENCES

[1] Gang Wang, Member, IEEE, David Forsyth, Fellow, IEEE, and Derek Hoiem, Member, IEEE, "Improved Object Categorization and Detection Using Comparative Object Similarity", IEEE Transactions on Pattern Analysis And Machine Intelligence.

[2] Gidudu Anthony, Hulley Greg and Marwala Tshilidzi, "Classification of Images Using Support Vector Machines",Department of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg.

[3]Rajkumar N.Satare and S. R. Khot, "Image matching with SIFT feature", Proceedings of the Second International Conference on Inventive Systems and Control.

[4] Heena R. Kher and Dr. Vishvjit K. Thakar, " Scale Invariant Feature Transform Based Image Matching and Registration ", Fifth International Conference on Signal and Image Processing, 2014.

[5] "SIFT: Theory and Practice ". [Online]. Available: http://aishack.in/tutorials /sift-scale-invariant-feature-transform- introduction/. [Accessed: 10- May- 2019]

[6] Chih-Chung Chan and Chih-Jen Lin, Support Vector Machines" , 2013. [Online]. Available: https://www.csie.ntu.edu.tw /cjlin /papers/libsvm.pdf

[7] " Introduction to OpenCV-Python Tutorials ".[Online]. Available: https://opencv-python -tutroals.readthedocs.io/en/latest/py tutorials /py setup/py intro/py intro.html. [Accessed: 10- May- 2019]

[8]David G. Lowe,"Distinctive Image Features from Scale-Invariant Keypoints ",2004. [Online]. Available: https://www.cs.ubc.ca/ lowe/papers/ijcv04.pdf