Advanced Agro Field & Crop Surveillance Systems

K.Nagarajan, K.Sumathi, Megha Nagarajan

Abstract: ***Agriculture becoming the major driver for Indian economy, applying some of the latest technological digital innovations to solve critical Agri-based challenges are becoming vital to improve the productivity and lower the cost of operations. Primary productivity index of agriculture is directly dependent on how much the crops escaped from attacks either by pests or by external intruders. Applying some of the advanced machine learning techniques in Computer Vision and multiple object detection algorithms in the field of Agriculture surveillance generates huge interest among farmer communities. In this paper, an aapproach which includes deployment of sensors to monitor the whole cultivation area, fixing appropriate cameras and detecting motions in the agro field, is proposed for Agro field surveillance. An orchestrated deployment of necessary sensing devices such as motion-sensing, capturing video based on demand and passes it on to the deep learning algorithms for further synthesis. The model is developed and trained leveraging technologies such as tensorflow, keras with google Colab, Jupyter notebook environment that runs entirely in the google cloud that requires very minimal setup. To evaluate the model, the authors create a test set which contains 200 captured events, more than 60,000 images that are relevant for this scope and available in public to train Deep Learning CNN based models.***

Keywords :Agro Field Monitoring, Crop Surveillance System, Applied Deep Learning Vision Algorithms & IoT Sensors, CNN Models

I. INTRODUCTION

Agro-Field & Crop Surveillance system is built on open source technologies both from the machine learning framework as well as leverage the existing convolutional neural network models that were trained on images such as Inception V5, YOLO V3, Resnet etc. Solution constitutes very robust data acquisition methods that are very efficient in real time, Transfer learning based Convolutional Neural Nets for accelerated learning and context specific learning that is needed in the field and a seamless orchestrated layer for effective decision making and perform appropriate actions. Keras is an Open Source Neural Network library [1] written in Python that runs on top of Tensorflow. Keras was developed by a Google engineer named François Chollet. It is very fast and easy to use due to it's greater level of abstraction. Keras is high-level API act as a wrapper for the low-level API that is capable of running on top of CNTK, Theano and TensorFlow.

Using Keras high level API, new models can be built, layers of the model can be defined and multiple input and output models can be set up. In Keras, Compilation will be done with loss and optimizer functions, training process will be done with fit function. Low level functions such as building tensors and variables, computational graph will be handled by backend engine which will perform the computation and necessary aids for model developments. Keras uses Tensorflow as a default "backend engine" that can be changed in the configuration.

Tensorflow is a low level API which enables the user to make an arbitrary computational graph or model layers. Tensorflow is a matured framework in the field of deep learning, developed by Google's Brain team it is the most popular deep learning tool.

While leveraging universal computer vision models, it is very important to understand the coverage that models have and also the relevance of agriculture contexts to be understood. A Brief study on image coverage be done before deciding on the models that are leveraged for Transfer Learning. This approach helps both in terms of object detection accuracy in the image as well as in the video and reduce the training time of these models.

An Orchestration layer to synthesis all deep learning vision algorithm outputs is the vital layer to make right decision and also trigger events as needed. These actions are contextual in nature, based on certain conditions, time of day, type of crops etc. It will have a nice feedback mechanism to learn and correct as time goes. A nice visual display on an App is the end user computing mechanism through which the entire solution would get consumed by the farmers and ecosystem players.

II. RELATED WORKS

NeuroHive [2], proposed a system for Automated Animal Identification Using Deep Learning Techniques. They applied Deep Learning algorithms to identify the presence of animals, to identify which animal is present, counting the number of animals**,** describing additional animal attributes count and described the behavior of animal species. Different classifiers are used to predict the accuracy and suggested that with VGG is the best model yielding 96.8% of accuracy.

Hamza Bendemra[3] presents techniques to make steps towards developing an algorithm that could be used for a classic image classification problem. He has implemented source code which will accept image submitted by user as input and returns an estimate of the dog's breed. The algorithm

Revised Manuscript Received on December 05, 2019.

Mr.K.Nagarajan,Principal Consultant, TCS, Chennai, Tamilnadu nag12317@gmail.com

Dr.K.SUMATHI, Assistant Professor, Department of CS & IT, Kalasalingam Academy of Research and Education, Krishnankoil, Virudhunagar, Tamilnadu, India,.Sumathirajkumar2006@gmail.com

Ms.Megha Nagarajan,II year BE(CSE) St.Joseph's College of Engineering, Chennai, Tamilnadu meghanagarajan72@gmail.com

returns an estimate of the dog breed that is most resembling if it detects human.

Alexandra Swanson et al., [4] established the Snapshot Serengeti camera survey to assess spatial and temporal dynamics of large predators and their prey. They have executed a camera survey expands upon historical monitoring by providing the first continuous systematic data on all of the larger predator and prey species, day and night, across several years. The camera-trap grid offers systematic coverage of the entire study area and ensures at least two cameras per home range for each medium to large mammalian species.

Alex Krizhevsky et al [5] trained a large, deep convolutional neural network to classify the high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, they achieved top-1 and top-5 error rates of 37.5%and 17.0%. They used non-saturating neurons and a very efficient GPU implementation of the convolution operation to make training faster. They employed a recently developed regularization method called "dropout" that proved to be very effective to reduce over-fitting in the fully-connected layers.

III. PROPOSED FRAMEWORK

Major motivation for crop surveillance system is to identify opportunities in the field and solve it through latest digital expertise. Multiple objects detection is the primary deep learning vision techniques to identify different parties involved in the scene, thereby configurable alerts can be triggered based on detection and scene intent.

The proposed method obtains the object in the pictures from motion detection camera mounted on field and various sensing IoT devices to monitor and correlate events. The role of motion detection camera is to record the movement of an object and their visual appearance. The motion detection camera saves video when it detects movement. The motion includes time lapse setting for regular monitoring. The surveillance images or videos are recorded and captured images are inputs to deep learning algorithms which detect all interested objects on each scene and get it orchestrated for next best actions.

Solution construct for this challenge constitutes four components

1) Survey of the whole cultivation area with appropriate boundary mapping and identify elegant spots for optimal coverage of the field to deploy sensors and cameras.
2) Placing appropriate cameras with reasonable pixel size to capture video based on trigger points, and gets orchestrated from the deployed sensors to detect motions in the agri field
3) Transferring captured images and videos to common location for further analysis with appropriate time and event tags
4) Transfer learning from pre-existing models that had past learning of objected to be trained for contextual images and inputs and Execution of Deep learning algorithms to detect multiple objects and triggering appropriate actions based on scenarios and objects found on the scene.

Deployment of right sensors at right places followed by optimal triaging procedure for scene to capture is the important aspects of scene acquisition. The Data Flow Diagram of the proposed approach is given in figure 1.

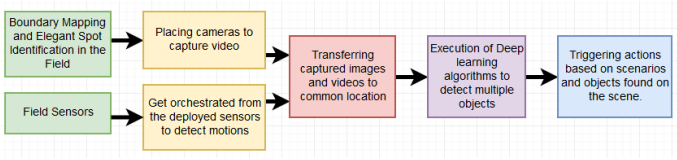
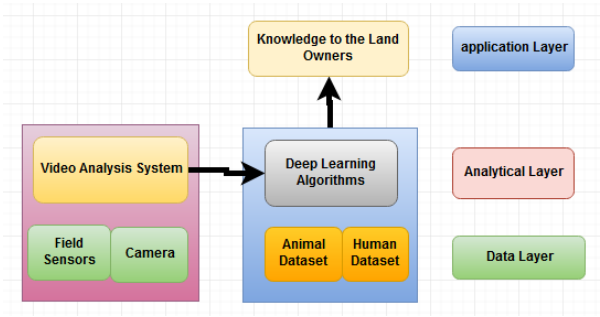


Figure 1. Agro Field & Crop Surveillance Systems - DFD

Model training with various field captured images that have pests and external intruders through their labels and tags. Auto-tagging tools are leveraged for faster tagging and thereby improve the overall training time.

The anticipated accuracy for these models has to be reasonably high, transfer learning are adopted from ImageNet and RestNet based on different scenarios. The component view of the Agro Field & Crop Surveillance Systems is proposed in Figure2.

Figure 2 – Components of Agro Field & Crop Surveillance Systems



Power source for the entire solution designed to be optimal, which were enabled through solar panels and standard power backup for night time surveillance. All these entire solution components are enclosed in a waterproof wrapper supported by anti-theft alarm inside. Complete Data Life Cycle Management are deployed to preserve vital field signatures and remove standard recordings that are deemed inappropriate.

Mobile application will be developed to report incidents such as pest attacks, right classification of pests, external intrusion on the fields and alerts in real time to farmer communities and also sharing the intelligence to any nearby farmers based on subscriptions of events. Status checks for each of the devices will be autonomous operations

The entire solution could have a significant impact among the farmers, to protect their Field and Crop seamlessly 7 X 24 all days of the year.

This requires a very minimal maintenance and a quarterly recalibration of deep learning object detection models along with providing additional intelligence around new pests and external intruders of the field.

Train CNN Model

CNN model is built to classify images of human and animals like rat, horse, elephant, bear, snake, pig, rabbit, peacock, etc. animal dataset is collected from internet. The dataset contains both training and test data. A high level API named "Keras" is used to built and train the model in tensorflow. The important step in model building is the collecting the image with labeling. Google Colab notebook is used for model building, evaluation and prediction. In the first step of implementation, TensorFlow, numpy, matplotlib and Keras packages are imported. The implementation includes explore the animal and human datasets, built the CNN classification model, evaluate the training and validation accuracy. The images and video streams captured from the field camera are uploaded in cloud. This paper presents an image classification model that is designed by preceding the following steps. The model building process can be divided in to five stages.

1) Prepare the training contains images with corresponding labels and test dataset contains only images 2) Loading and preprocessing 3) Design a model for image classification 4) Train the model with test dataset 5) Estimate and enhance the performance

The important step in model building process are defining the a) number of convolutional layers in the model b) activation function for each layer c) number of hidden units in each layers. Once model is constructed, it can be tested using validation dataset which is a part of training dataset so that we can see the performance of the proposed model on

unseen data before exploring it to the test dataset. The proposed model can be trained using training data and validated using validation data.

The images contents of the .zip are extracted to the base directory `/tmp/mydataset', which contains `train` and `validation` subdirectories for the training and validation datasets which in turn each contain 'elephant', 'horse','cats` ,'cow', `dogs` ,etc., subdirectories. Let's define each of these directories:

total training cat images: 172
total training cow images: 195
total training horse images: 601
total training dog images: 261
total training elephant images: 371
.....
.....
total validataing cat images: 53
total validating cow images: 55
total validation horse images: 50
total validation dog images: 33
total validation elephant images: 43
.....
.....

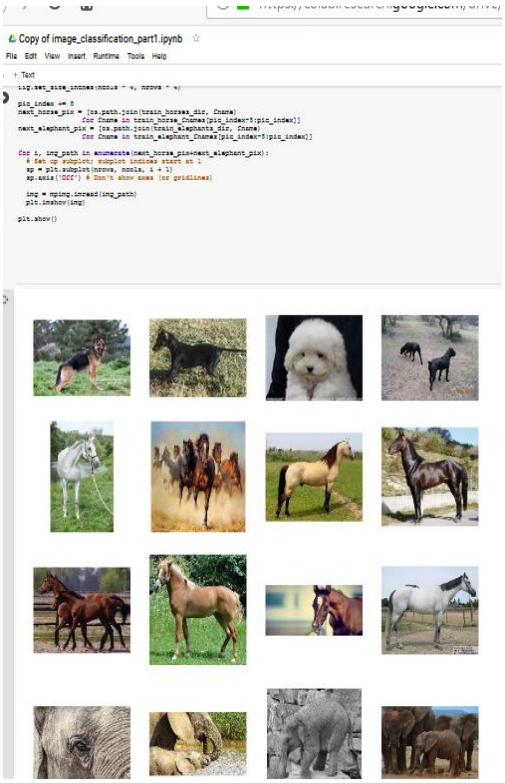


Figure 2a – Batch of Animals

Here a two-class classification problem (binary classification with sigmoid activation) is used and the output of this model will be a single scalar between 0 and 1 and encoding the probability that the current image is class 1

```
model.summary()

Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 150, 150, 3)]     0
_________________________________________________________________
conv2d (Conv2D)              (None, 148, 148, 16)      448
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 74, 74, 16)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 72, 72, 32)        4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 36, 36, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 34, 34, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 17, 17, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 18496)             0
_________________________________________________________________
dense (Dense)                (None, 512)               9470464
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 513
=================================================================
Total params: 9,494,561
Trainable params: 9,494,561
Non-trainable params: 0
```

Figure 2b – Model Architecture

The output clearly signifies each layers of influence wrt features considered, number layers considered yield reasonably good results. Appropriate activation functions could also be tried for false positive scenarios, exclusive considerations and mathematical functions are to be evaluated in the future work

The model can be used to make predictions on the test data set, once its performance is fine on the validation dataset. Model Loss, Model Accuracy, Time to generate

model(Individual epochs) is shown in Fig 3.

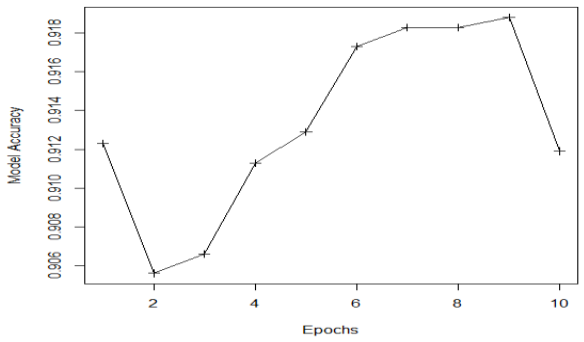


Figure 3a – Model Accuracy(individual epochs)

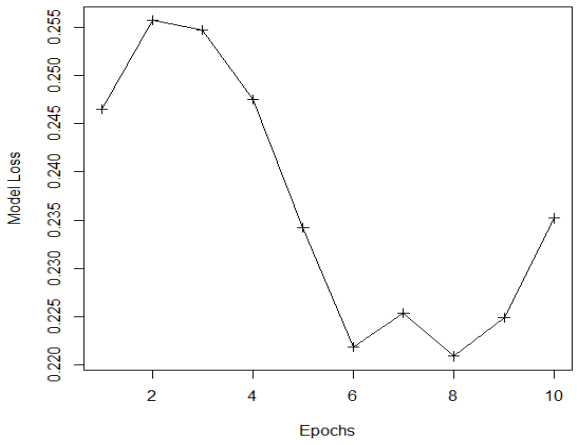


Figure 3b – Model Loss (individual epochs)

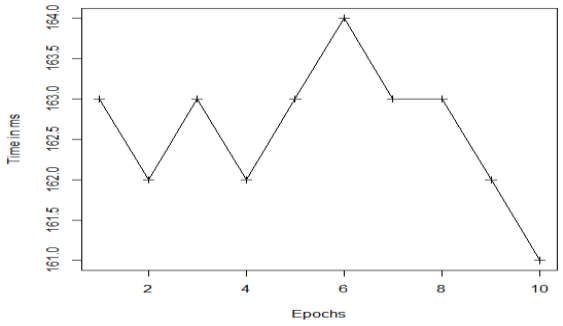


Figure 3c – Model Generation Time (individual epochs)

Model is trained using Training images and their corresponding true labels and Validation images and their corresponding true labels. Number of epochs should be defined in this step. At last test data set is loaded and preprocessed, and then predicts the classes for these images using the trained model.

IV. CONCLUSION

The entire solution results and value depends on the accuracy of the deep learning vision algorithms as well as the speed of achieving this solution. Therefore it's very important to experiment Transfer Learning aspects as much as possible, collection of specific scenarios on the fields and entities, orchestrated execution of events and actions. Accuracy improvements and real time streaming with multiple interested object detection would be the key for this solution where we will have a clear baseline and show improvements as we learn more and get differential contexts on the field.

FUTURE WORK

Immediate next step on this CNN model, results to be compared with Transfer learning models built out from Imagenet. The work can be extended for real-time object detection using deep learning and OpenCV to work with video streams and video files. This will be done using the highly efficient Video Stream. The deep learning real-time object detector can be built using YOLO and OpenCV. Video stream may be preprocessed in an efficient manner and, applying object detection to each frame.

REFERENCES

1. https://www.guru99.com/keras-tutorial.html
2. NeuroHive, Data science state-of-the-art: neural networks, machine learning, computer vision, https://medium.com/neurohive-computer-vision
3. Hamza Bendemra, Editorial Associate at Towards Data Science | Machine Learning Engineer, http://www.hamza-bendemra.com/
4. Alexandra Swanson, Margaret Kosmala, Chris Lintott, Robert Simpson, Arfon Smith & Craig Packer, Snapshot Serengeti, high-frequency annotated camera trap images of 40 mammalian species in an African savanna, Scientific Data volume 2, Article number: 150026 (2015)
5. Alex Krizhevsky, IIya Sutskever, Geoffrey E. Hinto, ImageNet Classification with Deep ConvolutionalNeural Network,http://code.google.com/p/cuda-convnet
6. https://www.edureka.co/blog/tensorflow-object-detection-tutorial/
7. https://towardsdatascience.com/image-recognition-with-keras-convolutional-neural-networks-e2af10a10114
8. https://www.guru99.com/download-install-tensorflow.html
9. https://www.analyticsvidhya.com/blog/2019/01/build-image-classification-model-10-minutes/