An IoT based system to identify and notify the user about the ripening stage of the banana

Omkar Vivek Sabnis
School of Computer Science and Engineering,
Vellore Institute of Technology,
Vellore, India.
omkarsabnis79@gmail.com

Gana Teja Akula
School of Computer Science and Engineering,
Vellore Institute of Technology
Vellore, India.
akulaganateja@gmail.com

Kevin Abraham
School of Computer Science and Engineering,
Vellore Institute of Technology,
Vellore, India.
kevinabraham335@gmail.com

Adheik Dominic
School of Computer Science and Engineering,
Vellore Institute of Technology
Vellore, India.
adheikdominic09@gmail.com

***Abstract* – As the world becomes more machine learning oriented, it can be seen that many of the jobs done by humans will be done by robots. The robotic revolution will be infinite from the most complex tasks such as surgeries to the simplest task such as moving objects, everything will be done by robots. Machines have already been integrated into our world. Companies are using robots to identify parcels and other deliveries. Amazon wants to use image processing, machine learning and drone technologies to control this system. As engineers of the future generation, we decided to employ these technologies to try and emulate the technology on a smaller scale. We decided to make this system using image processing for analyzing if the banana is ripe or unripe. We used machine learning in the form of clustering to train the system based on a dataset we collected of different pictures of different bananas at different ripening stages and we deployed our model on the cloud for processing and used the cloud services to notify the users of the ripeness.**

Keywords— Object Detection, Machine Learning, MobileNet V1, Indian Agriculture, Banana Ripening.

I. INTRODUCTION

A perfect ripening process is one in which the crop is neither over ripened nor under ripened. For the purpose of successful ripening of a fruit, the various ripening stages have to be monitored carefully so that the fruit is not wasted. This has to be done with special care as agriculture in India serves the basic principle of livelihood to its citizens. A farmers' life depends only on successful agriculture. In this new era of science and technology farmers have also started using various technologies to improve their crop production [1-5]. This paper presents various supervisory stages of ripening of banana using an Arduino. The ripening process of banana follows a specific pattern which involves various stages [6]. To begin with, banana in their primary stage is completely green. Next, they turn yellow with their tips being green. In a more mature stage, they turn completely yellow. After turning wholly yellow, they start to get some brown spots which finally turn black. In this way the ripening process of banana is completed, which is shown by a figure 1 below.

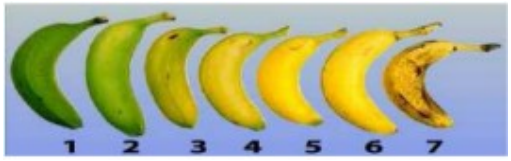


Fig. 1. Ripening stages of banana

II. LITERATURE SURVEY

1. Arduino Based Supervision of Banana Ripening Stages, 1st International IEEE Conference on Next Generation Computing Technologies (NGCT – 2015)

Challenges – The major challenge faced by the authors was image acquisition. The authors were using an external sensor, that only sent RGB values to the Arduino, thereby reducing the load on the microcontroller. However, since the image wasn't recorded and there was no segmentation done, the process requires the sensor to be brought very close to the fruit, reducing its application in an automated environment.

Methodology – The paper follows the process of supervising the various ripening stages by classifying them into 7 stages. The sensor will send RGB values using a MATLAB code and the information is sent to the Arduino which decides the ripening stage and informs the user using GSM.

Applications – The authors of the paper aim to integrate technology with agriculture. The aimed to create a cheap and easy method to automate the checking the ripening stages of bananas and fruits that can be used without any presence in the field.

Pros and Cons – The pros of the system are that it is easy to use and cheap to build. The cons of the system are that it doesn't acquire any image of the banana, using just a sensor to read the RGB of the object in front of the sensor.

2. Ripeness Classification of Bananas Using an Artificial Neural Network, Arabian Journal for Science and Engineering by Springer

Challenges – The major challenge faced by the paper is the method used is computationally expensive. Artificial

Neural Networks can take a lot of memory and using a microcontroller wasn't possible for them.

Methodology – The paper follows the process of collecting a database of labelled banana images, which can then be used for training. The proposed system and other machine learning algorithms are trained on the database and compared. The neural network system achieved an accuracy of 97.75% and the system performs accurately when fed a picture of a banana.

Applications – The paper has been made for industries, where computationally expensive models can be used. They plan on implementing this system in a computer vision machine for a food processing plant.

Pros and Cons – The pros of the system are that it has a very high accuracy and the speed of classification after training is very high. The cons of the system are that it can be used in industries and not in an embedded system. It is also very memory heavy.

3. Recognizing the Ripeness of Bananas using Artificial Neural Network based on Histogram Approach, 2009 IEEE International Conference on Signal and Image Processing Applications

Challenges – The major challenge faced by the paper was sample size. The neural network used is small, making it an efficient process however, the training set of images consisted of only 32 images while testing was performed on 28 images.

Methodology – The paper follows the process of collecting a database of labelled banana images, which can then be used for training. They trained a 3 layer fully connected neural network on 32 images captured from a 2 MP camera and then tested it on 28 images. They achieved an accuracy of 88% and the system was developed in MATLAB.

Applications – The paper has been made for industries, where computationally expensive models can be used. They plan on implementing this system in a computer vision machine for a food processing plant.

Pros and Cons – The pros of the system are that the network design is efficient and can be run on an embedded system. However, the cons are that the image quality was bad and the training sample size was too small – leading to lower accuracy.

4. Deep Indicator for fine-grained classification of banana's ripening stages, EURASIP Journal on Image and Video Processing by Springer

Challenges – The major challenge faced by the paper was using a method that can take into account the degradation of the banana and creating a novel architecture for the convolutional neural network.

Methodology – The paper follows the process of creating a novel convolutional neural network architecture. This network then learns from a set of fine grained image features other than skin color based on a data driven mechanism. The process gives an accuracy better than the state of the art performance at the time of publication.

Applications – The paper has been made for industries, where computationally expensive models can be used. They plan on implementing this system in a computer vision machine for a food processing plant.

Pros and Cons – The pros of the system are that the paper claims it is the first attempt to introduce deep learning to classification of bananas and their performance is the new state of the art. The cons of the system are that it is computationally expensive and difficult to reproduce in an outdoor system because the training samples are images in a controlled environment.

5. Classification of ripening stages of bananas based on Support Vector Machines, International Journal of Agricultural and Biological Engineering by Elsevier

Challenges – The major challenge faced by the paper was the difficulty in recognizing the ripening stages using the L*, A* and B* which is the CIESLAB color scheme.

Methodology – The paper follows the process of analyzing the peel color at 3 different locations on the banana, the stalk, the middle and the tip. Then, a support vector machine was used to classify the banana using the LAB values. The SVM used the 10-fold cross validation method of classification, reaching an accuracy of 96.5%

Applications – The paper didn't use any expensive equipment and the paper suggests that if the training of the SVM can be done on a system, the classification of the bananas can be performed on an embedded system.

Pros and Cons – The pros of the system are that the paper uses 3 different color parameters to find the ripening stages rather than just the general color of the banana. They also used a unique color space which showed significant trends to recognize the ripening stage of the banana. The cons are that it has high requirements on the training system and requires multiple images rather than using image segmentation for the 3 different parameters.

III. Proposed Approach

This part of the report will explain in detail the novel approach we are applying to find a solution to this problem. Convolutional Neural Networks can learn extremely complex mapping functions when trained on enough data. We can't yet understand how a convolutional net learns such complicated functions. [7]Basically the training of a CNN involves, finding of the right values on each of the filters so that an input image when passed through the multiple layers, activates certain neurons of the last layer so as to predict the correct class. Though training a CNN from scratch is possible for small projects, most applications require the training of very large CNN's and this as you guessed, takes extremely huge amounts of processed data and computational power. And both of these are not found so easily these days. That's where transfer learning comes into play. In transfer learning, we take the pre-trained weights of an already trained model (one that has been trained on millions of images belonging to 1000's of classes, on several high power GPU's for several days) and use these already learned features to predict new classes. The advantages of transfer learning are that there is no need of an extremely large training dataset, not much computational power is required. As we are using pre-trained weights and only have to learn the weights of the last few layers. Keeping in mind that we want to use transfer learning, we looked at various models like VGG16, VGG19 and MobileNetV1. We choose MobileNetV1 because of the following reasons; (1). It uses Depth Wise Separable Convolutions to reduce model size and complexity. It is the state of the art for mobile and embedded vision applications. (2). It uses much lesser

parameters for learning leading to the smaller model size. (3). It also has fewer multiplications and additions leading to lesser complexity. It has two main parameters than can be used for tuning the MobileNet to make it do what we need it to do – Width Multiplier α and Resolution Multiplier ρ.

Depth Wise Separable Convolution: It is a depth wise convolution followed by a pointwise convolution. [10] The depth wise convolution is a channel based spatial convolution – our images have 3 channels so we have 3 spatial convolutional operations. The pointwise convolution is a 1x1 convolution which is used to change the dimensions. Thus, when we compare this method to normal convolutions, we can see the following:

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F$$

Depthwise Separable Convolution Cost: Depthwise Convolution Cost (Left), Pointwise Convolution Cost (Right)

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F$$

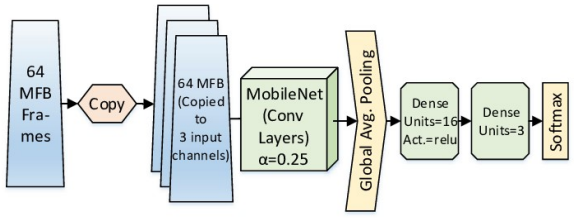
Standard Convolution Cost

In these equations – Dk is Kernel Size, DF is the feature map size, M is the input channel size and the N is the output channel size.

In one of our layers, we have the following scenarios – Dk = 3, DF = 5, M = 3 and N = 3. Hence, Depth Separable Convolution costs = 900 and Standard Convolution costs = 2025. This allows us to have less computations and have smaller drops in accuracy. Comparison to other State of the Art architectures:

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

MobileNet Comparison



MobileNet Architecture

TensorflowJS: TensorFlow.js is a JavaScript library developed by Google for training and using machine learning (ML) models in the browser. It's a companion library to Tensorflow, a popular ML library for Python. Some of the features of this library are:

Speed - TensorFlow.js is hardware-accelerated because it uses WebGL (a JavaScript graphics API), so it has surprisingly good performance. A Node.js version of Tensorflow, tfjs-node, also exists and offers improved performance over the browser version.

Load existing models - One of the most important features of TensorFlow.js is that it allows you to load pretrained models. That means you can use libraries like this one and include image classification and pose detection on your website without the need to train the model yourself. TensorFlow.js also allows you to load models you've trained in the Python version of Tensorflow. That means you can write a model and train it using Python, then save it to a location available on the web and load it in your JS. This technique can significantly improve performance because you don't have to train the model in the browser.

Some of the use cases of this library are:

More and more, businesses are using machine learning to improve interactions with users. AI programs handle everything from self-driving cars to matchmaking in video games, chatbots like Siri and Alexa, and suggesting content for users. In the past, however, machine learning has been handled on back-end servers.

The creation of TensorFlow.js means that you can create and run AI models in a static HTML document. Yes, you heard that right: you can use AI without setting up a server or even a database. As long as the user's browser supports JavaScript (and preferably WebGL) you can train and use ML models, all client-side.

Here are some uses of ML (not all examples use TensorFlow.js) to fill your mind with possibilities:

Create abstract art: Though this example is less "useful" for the real world (unless you want to become an art dealer), this is one of my favorite examples

Generate realistic images: thispersondoesnotexist.com recently made the news for using a generative adversarial network to generate images of completely new people. This website explains how a neural network developed by Google "finds" objects in unrelated images.

Play games: Having AI players in video games isn't a new idea, and there are already examples in TensorFlow.js. A project uses TensorFlow.js to automate the Chrome Dinosaur game.

Recommend content: Content recommendation through AI is fairly popular and used by most media platforms. With TensorFlow.js, content recommendation can be handled on the client side

Cloud Processing – Heroku:

Create an app on Heroku, which prepares Heroku to receive your source code.

```
$ heroku create

Creating sharp-rain-871... done, stack is heroku-18

http://sharp-rain-871.herokuapp.com/ |
https://git.heroku.com/sharp-rain-871.git

Git remote heroku added
```

When you create an app, a git remote (called heroku) is also created and associated with your local git repository.
Heroku generates a random name (in this case sharp-rain-871) for your app, or you can pass a parameter to specify your own app name.
Now deploy your code:

```
$ git push heroku master

Counting objects: 488, done.

Delta compression using up to 8 threads.

Compressing objects: 100% (367/367), done.

Writing objects: 100% (488/488), 231.85 KiB | 115.92 MiB/s, done.

Total 488 (delta 86), reused 488 (delta 86)

remote: Compressing source files... done.

remote: Building source:

remote: Verifying deploy... done.

To https://git.heroku.com/nameless-savannah-4829.git

 * [new branch]      master -> master
```

The application is now deployed. Ensure that at least one instance of the app is running:

```
$ heroku ps:scale web=1
```

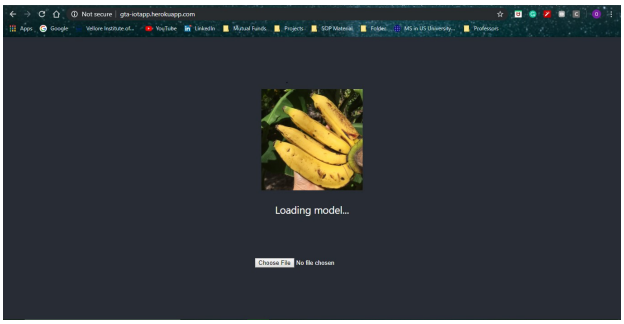
Now visit the app at the URL generated by its app name. As a handy shortcut, you can open the website as follows:

```
$ heroku open
```

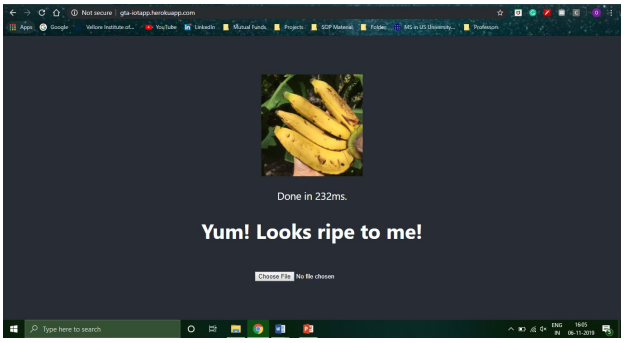
Thus, we have deployed our TensorflowJS application on the cloud on the Heroku Server.
Convolutional Neural Networks: A convolutional neural network is an artificial neural network that has so far been the most useful in analyzing images. They are used to detect patterns in an image, using the hidden convolutional layers. Basically, each of these layer consist of filters, which help highlight certain portions of the image and can then be used to figure out certain patterns [15]. For example, it can be used to highlight edges and deeper the convolutional layers go, it can be used to detect more complex patterns, like faces. Some of the important features of Keras and Convolutional Neural Networks are:
Sequential Model – A model defined by keras that helps define a linear stack of layers. This helps get the basic structure of the neural network in place.
Flatten, Dense, Dropout – Flatten layer is used to convert a 2 dimensional array of vectors into a single long layer of vectors. This is needed by adding a dense layer as fully connectivity requires a flattened vector. Dense Layer is a fully connected layer. It is usually used for combining all the features acquired from the previous layers. Dropout layer is a regularization method to reduce overfitting. Overfitting means when a model overcomplicates to explain even the outliers. This layer removes certain hidden and visible units in the network. It is a way of averaging the model.
Maxpooling2D and Zeropadding2D – Max Pooling is an operation in which we move a 2x2 window across the image tensor and select the maximum of the values within that window [13]. This achieves two major objectives – one is reducing the parameter size and second is generalizing the results so features extracted will not change with the scale and orientation of the image. Maxpooling2D means for a 2 dimensional data there is a 3D and 1D as well. Zeropadding2D is the layer that adds zero padding to the image tensor.
Convolution2D – A layer is used for convolution. Convolution is a process of running a filter of size x over the image and doing the element-wise multiplication operations on it. This leads to the image getting smaller as the multiplications are then summed up to bring about a single number. This is done to the whole image and we get an activation map at the end of this process.
Numpy – A very powerful python package that is used for simplifying the matrix operations such as addition, multiplication and so on. It is also useful in converting data into a matrix form.
Function POP – This is used to remove the layers of a network. This helps us because our network needs the dense and dropout layers for training and arriving at a form where we have a matrix of weights. This will then be stored as the weights file. However, as we want to extract the features, we must remove those last 2 layers so that we can get the 4096D activations. We want those activations as they will represent the activated features
Load_model_legacy – This is a function that is used to load and store the weights file. This is also used as stop-gap arrangement with the various versions of keras on various systems. The weights were trained in a legacy version of keras and some future versions of keras don't accept this format. Hence, this function helps in loading the weights file irrespective of the keras versions.
The network – We initialize the sequential model so that we can linearly arrange the network layers. The first layer is a padding layer that will add zero padding to the image tensor. We then use a convolutional layer with a filter of size 3x3 with the activation function 'relu'. There are 4 main types of activation functions – 'sigmoid', 'tanh', 'relu' and 'leaky relu'

IV. Results

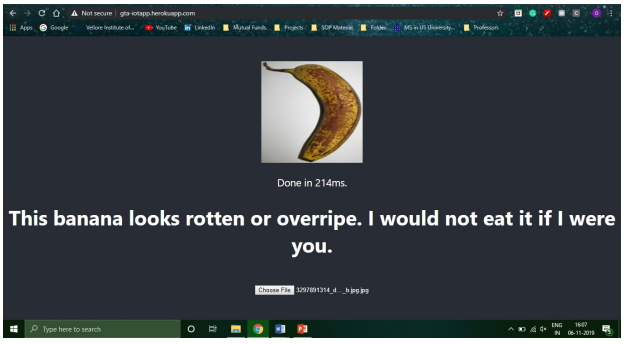
The next section describes the various stages of our project
Step 1 – Loading the model on our cloud server



Model being loaded on the cloud – On deployment



Model Deploys on the Cloud – making primary prediction



Prediction of the model on the Cloud – using user input image

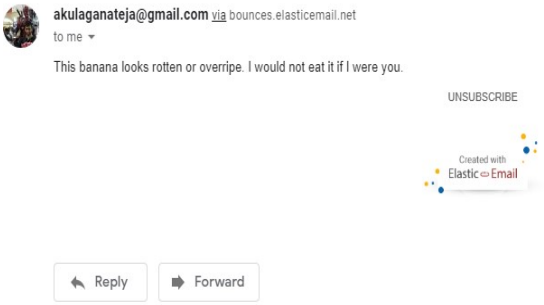
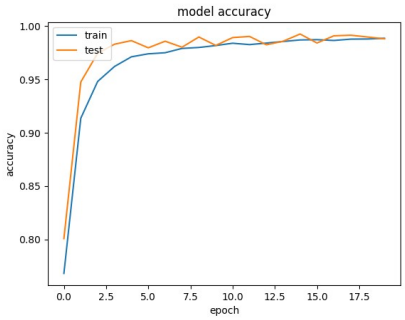
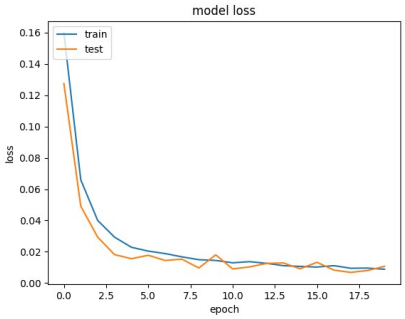


Image received on user's email



Model Accuracy Graph



Model Loss Graph

We achieved a model accuracy of 97% on training and a validation accuracy of 87.4%. Our testing accuracy was 96.58%. Our model training loss was 0.412.

V. Conclusions

This system is specifically designed to integrate technology with agriculture. This is a smart combination of different devices to ease the farmer about the different stages of ripening of crops without any physical presence in the field. The technique can also be used in the cold storage or fields to avoid the spoilage of crop due to over or under ripening. The system is simple and low cost and easy to manufacture and can be used in other crops as well.

VI. Future Work

As the future scope of this research, we will analyze the various algorithms that can be used for identification of the fruits and their ripening stages and do a comparative analysis of which algorithms perform better. The system proposed will be deployed in an embedded setting so we will do a comparison while keeping parameters like accuracy for object detection as well as the frame-rates which the algorithm provides.

Acknowledgment

The authors would like to thank all the reviewers and experts who have supported our research and provided us with their valuable insights and thoughts which helped us improve the present research work.

References

[I] Shen Jin, Song Jingling, Han Qiuyan, Wang Shengde, YangYan, School of Electric and Electronic Engineering, A Remote Measurement and

Control System for Greenhouse Based on GSM-SMS IEEE 8th International Conference on Electronic Measurement and Instrument, 2007.
[2] M. Kass, A. Witkin, and D. Terzopoulos, Snakes-active contour models, International Journal of Computer Vision, vol. I, pp. 321331, 1987
[3] Yuwei Wu, Yuanquan Wang and Yude Jia, Adaptive diffusion flow active contours, International Journal of Computer Vision, 2013, pp. 14211435.
[4] Technical Centre for Agricultural and Rural Cooperation (CTA), Routine Post-Harvest Screening of BananaJPlantain Hybrids: Criteria and Methods, 1997
[5] H. Saadl , A. Hussain 2, Ripeness classification of papaya by using artificial neural and threshold method, 2006.
[6] Daniel K. Fisher and Hirut Kebede A low-cost microcontroller-based system to monitor crop temperature and water status, Computers and Electronics in Agriculture, Elsevier B. V. Electronic Measurement and Instrument, 2007 .
[7] J.B. George, B.M. Mwangangi Some Factors Affecting Banana Storage And Ripening: A Case Study Of Banana Handling And Ripening In Kenya International Symposium on Postharvest Treatment of Horticultural Crops 1994.
[8] Simon monk Programming Arduino Getting Started With Sketches by Tata McGraw Hill.
[9] Sourangsu Banerji, Design and Implementation of an Unmanned Vehicle using a GSM Network with Microcontrollers, International Journal of Science, Engineering Technology Research, Vol.2, Issue.2, 2013.
[10] Arney Kelkar Implementation of Unmanned Vehicle using GSM Network with Arduino International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 4, April 2014.
[11] S. Sumriddetchkajorn and Y. Intaravanne, "Data-nonintrusive photonicsbased credit card verifier with a low noise rejection rate," Appl. Opt., vol. 49, pp. 764-771, 2010.
[12] K. Suwansukho, S. Sumriddetchkajorn, and P. Buranasiri, "Demonstration of a single-wavelength spectral-imaging-based Thai jasmine rice identification," Appl. Opt., vol. 50, pp. 4024-4030, 2011.
[13] P. Rajkumar, N. Wang, G. EImasry, G. S. V. Raghavan, and Y. Gariepy, "Studies on banana fruit quality and maturity stages using hyperspectral imaging," J. Food Eng., vol. 108, pp. 194–200, 2012.
[14] F. Mendoza, "Characterization of surface appearance and color in some fruits and vegetables by image analysis," Ph.D. Thesis, Pontificia Universidad Católica de Chile, 2005.
[15] H. Saad, A. P. Ismaie, N. Othman, M. H. Jusoh, N. F. Naim, and N. A. Ahmad, "Recognizing the ripeness of bananas using artificial neural network based on histogram approach," Proc. of IEEE Conference on Signal and Image Processing Applications, pp. 536 – 541, 2009.