IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

# "PREVENTION OF FOOD POISONING BY DETECTING FRUIT QUALITY USING DEEP LEARNING"

[1]Prof Mamatha M, [2] Hrishikesh R , [3]Harsha Vardhan Pal, [4] Chiranjeevi Sagar K

[1]Professsor of Computer Science and Engineering, Brindavan College of Engineering
[2,3,4] Student of Computer Science and Engineering, Brindavan College of Engineering

***Abstract:*** **In a variety of industrial contexts, including factories, supermarkets, and other places, classifying fruits is crucial. People who use fruit classification to select the appropriate fruits may also benefit from it if they have special nutritional needs. Fruit classification used to be done manually, which takes time and necessitates a constant human presence. Machine learning methods for fruit classification have been proposed in the past in large numbers. Deep learning's detection and classification capabilities suggest that it could be a potent engine for producing results that can be applied to the world of today. The project presents a comprehensive analysis of a variety of fruit images for freshness grading using deep learning. A number of algorithms have been reviewed in this project, including YOLO for detecting region of interest with considerations of digital images, ResNet, VGG, Google Net, and AlexNet as the base networks for freshness grading feature extraction. Fruit decaying occurs in a gradual manner, this characteristic is included for freshness grading by interpreting chronologically-related fruit decaying information. The contribution of this proposed system is to propose a novel neural network structure, i.e., YOLO + Regression CNNs for fruit object locating, classification, and freshness grading. Fruits as an object, its images are fed into YOLO for segmentation and regression, then for freshness grading.**

***Index Terms*** **– Fruit quality, Fruit classification, CNN, YOLO, Regression.**

## I. INTRODUCTION

Fruit spoiling has a considerable impact on economic activity. It is estimated that close to one-third of the cost of fruit is spent on decomposing items. Decreased concentrations of amino acids, vitamins, sugar/glucose, and other nutrients inevitably raise public concerns about edibility issues, which all together prompt discussions on this subject to stop or slow down the decaying process. As a result, consumers believe that spoiled fruits are unhealthy.

Fruit rotting refers to how people judge the quality of fruit, including whether it is desirable, acceptable to eat, and disliked for having unpleasant sensory qualities (Akinmusire, 2011). According to research, there is a direct correlation between bacteria and fruit spoilage. This includes aerobic psychrotrophic gram-negative bacteria, heterofermentative lactobacilli, spore-forming bacteria, yeasts, and moulds (which secrete extracellular hydrolytic enzymes that damage plant cell walls).

Fruit degenerationn brought on by bacteria is a result of pectindegradation: a structural acidic heteropolysaccharide found in the cell walls of terrestrial plants and primarily composed of galacturonic acid.

## II. LITERATURE SURVEY

Before starting with the analysis and elegance of project, we've got to refer many analysis papers, manuals, documents associated with the thought of project. There are many paper concerns about fruit quality detection but many etiquettes.

A Machine learning Approach for Predicting Fruit Freshness Classification (IRJET) 2021.

Non-intrusive image processing Thompson orange grading methods.(IEEE)2017

Fruit recognition and maturity monitoring system.(IJCRT) 2021

These above mentioned systems are great for references .However, none of them used both deep learning approach and IOT together for quality detection.

## III. SOFTWARE IMPLEMENTATION

This work summarizes review of previous works that have done on fruit quality detection for prevention of food poisoning and related topics. Following are different methods and techniques used.

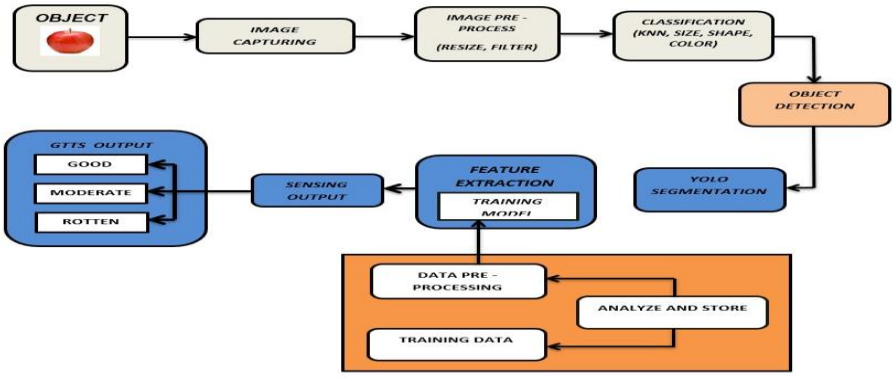


**Fig. 1 System architecture**

### 3.1 IMAGE PRE-PROCESSING AND CLASSIFICATION

Image processing libraries such as OpenCV and scikit-image represent RGB images as multidimensional NumPy arrays with shape (height, width, depth). This is done using matrix:

Defining the dimensions of matrix : we always write it as rows x columns. The number of rows in an image is its height whereas the number of columns is the image's width. The depth will still remain the depth. Therefore, while it may be slightly confusing to see the .shape of a NumPy array represented as (height, width, depth), this representation actually makes intuitive sense when considering how a matrix is constructed and annotated.

#### A. RGB and BGR Ordering

It's important to note that OpenCV stores RGB channels in reverse order. While we normally think in terms of Red, Green, and Blue, OpenCV actually stores the pixel values in Blue, Green, Red order. OpenCV library chose the BGR color format because the BGR ordering was popular among camera manufacturers and other software developers at the time .Simply put – this BGR ordering was made for historical reasons and a choice that we now haveto live with. It's a small caveat, but an important one to keep in mind when working with OpenCV.

#### B. Scaling and Aspect Ratios

Scaling, or simply resizing, is the process of increasing or decreasing the size of an image in terms of width and height. When resizing an image, it's important to keep in mind the aspect ratio.

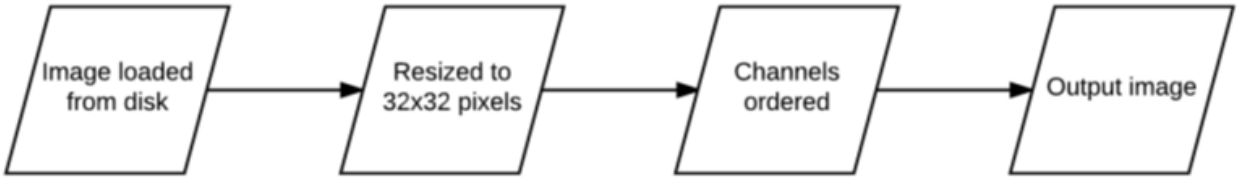


Image pre-processing pipeline that loads an image from disk, resizes it to 32_32 pixels, orders the channel dimensions, and outputs the image.

### 3.2 Convolutional Neural Networks

Each layer in a CNN applies a different set of filters, typically hundreds or thousands of them, and combines the results feeding the output into the next layer in the network. During training, a CNN automatically learns the values for these filters.

In the context of image classification, our CNN may learn to:

- Detect edges from raw pixel data in the first layer.
- Use these edges to detect shapes (i.e., "blobs") in the second layer.
- Use these shapes to detect higher-level features such as facial structures, parts of a car, etc.in the highest layers of the network.

The last layer in a CNN uses these higher-level features to make predictions regarding the Contents of the image.

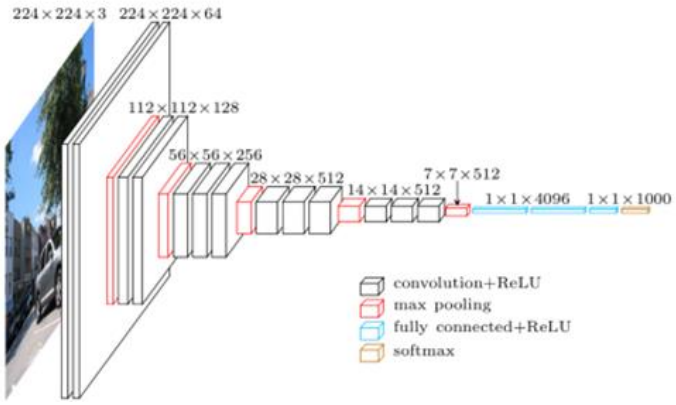


**Fig. 2 CNN Architecture**

In terms of deep learning, an (image) convolution is an element-wise multiplication of two matrices followed by a sum.

1. Take two matrices (which both have the same dimensions).
2. Multiply them, element-by-element (i.e., not the dot product, just a simple multiplication).
3. Sum the elements together.

## IV. HARDWARE IMPLEMENTATION

### 4.1 COMPONENTS USED :

#### A. ESP8266 – WIFI – Micro controller

**Fig. 3 ESP8266 – WIFI – Micro controller**

The ESP8266 WiFi Module is a self contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your WiFi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware, meaning, you can simply hook this up to your Arduino device and get about as much WiFi-ability as a WiFi Shield offers (and that's just out of the box)! The ESP8266 module is an extremely cost effective board with a huge, and ever growing, community.

#### B.MQ 7 Gas Sensor


**Fig. 4 MQ7 Gas Sensor**

MQ-7 is a Carbon Monoxide (CO) sensor, suitable for sensing Carbon Monoxide concentrations(PPM) in the air. The MQ-7 sensor can measure CO concentrations ranging from 20 to 2000ppm. This sensor has a high sensitivity and fast response time. The sensor's output is an analog resistance. The drive circuit is very simple , just a voltage divider; all you need to do is power the heater coil with 5V DC or AC , add a load resistance, and connect the output to an ADC or a simple OPAMP comparator. This sensor comes in a package similar to our MQ-3 alcohol sensor, and can be used with the rhydoLABZ breakout board.

**C.Moisture sensor**

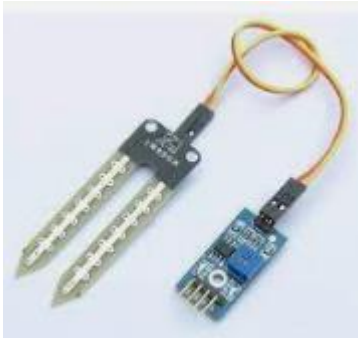

**Fig. 5 Moisture Sensor**

Moisture sensors measure the water content in the soil, fruits and other food items and can be used to estimate the amount of stored water. Moisture sensors do not measure water in directly. Instead, they measure changes in some other property that is related to water content in a predictable way.

## V. WORKING

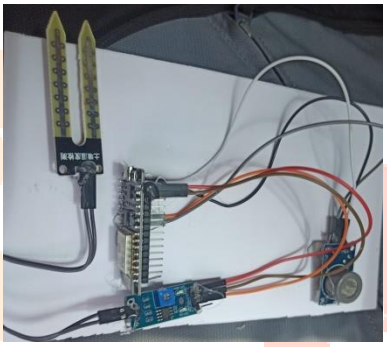

**Fig. 6 Sensor Connections**


**Fig. 7 Blynk App UI**

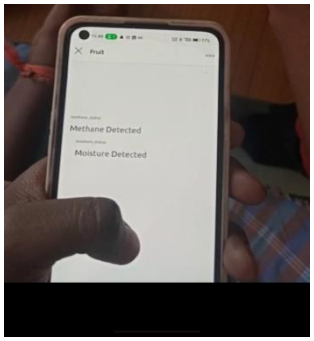


**Fig. 7 Sensor output on Blynk App**

The circuit uses an ESP8266 microcontroller to monitor the presence of methane gas and moisture, and send the data to the Blynk cloud platform using WiFi. Here is a step-by-step explanation of how the circuit works:

1. Power is supplied to the ESP8266 microcontroller using a 3.3V power source connected to the VCC and GND pins of the microcontroller.

2. The methane sensor and the moisture sensor are connected to the GPIO pins of the ESP8266. The methane sensor is connected to GPIO pin D5 and the soil moisture sensor is connected to GPIO pin D6.

3. The Blynk cloud platform is used to receive the sensor data from the ESP8266. The Blynk library is included in the code using #include <BlynkSimpleEsp8266.h> and the Blynk.begin() function is called in the setup() function using the authentication token, SSID and password of the WiFi network.

4. The sen() function is called every second using the BlynkTimer. This function reads the digital values of the methane and soil moisture sensors using the digitalRead() function and assigns the values to the variables methaneval and moistureval.

5. The function then sends the values of methaneval and moistureval to the Blynk cloud platform using the Blynk.virtualWrite() function and assigns the values to the virtual pins V0 and V1, respectively.

6. The function then checks the values of methaneval and moistureval. If the methane sensor detects the presence of methane gas, the function sends a notification message to the Blynk app using the Blynk.virtualWrite() function and assigns the message to the virtual pin V2. Similarly, if the moisture sensor detects the absence of moisture, the function sends a notification message to the Blynk app using the Blynk.virtualWrite() function and assigns the message to the virtual pin V3.

7. The loop() function is called repeatedly to run the Blynk.run() function, which maintains the connection to the Blynk cloud platform. The timer.run() function is also called repeatedly to run the sen() function every second.

Overall, the circuit uses the ESP8266 microcontroller, methane sensor, moisture sensor, and Blynk cloud platform to monitor the presence of methane gas and moisture and send the data to a smartphone app or other device.


**Fig. 8 Fruit quality Output(Good) fruit**

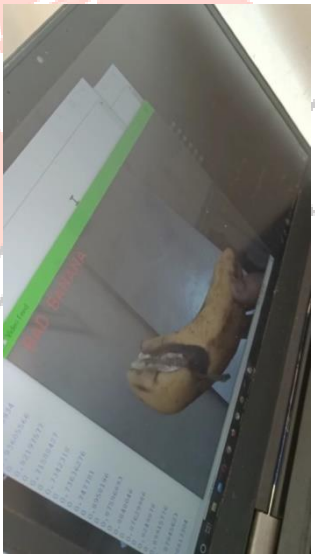

**Fig. 9 Fruit quality Output(Bad) fruit**

The system uses a pre-trained Convolutional Neural Network (CNN) model to classify fruits as either "GOOD" or "BAD". It uses the OpenCV library to capture video feed from a webcam and processes each frame by passing it through the pre-trained model to make a prediction. The predicted output is then displayed on the screen using the OpenCV library.

Here's a detailed step-by-step explanation:

1. Importing Libraries: The first step is to import all the required libraries such as os, keras, cv2, numpy, imutils, PIL, and gtts.

2. Loading the Model: The next step is to load the pre-trained CNN model 'fruits.h5' using the keras.models library.

3. Defining the Classes: The script creates a dictionary that maps each output class of the model to a fruit type.

4. Predicting Fruit Class: The script defines a function 'predict_class' that takes a frame from the video feed as input and processes it through the pre-trained model. The output of the model is a predicted class label for the fruit and a probability value. If the probability value is greater than 0.45, it means that the model has confidence in its prediction, and the predicted fruit type is displayed on the screen using the OpenCV library. If the probability value is less than 0.45, it means that the model is not confident in its prediction, and the script displays a message to the user to show the fruit more clearly.

5. Capturing Video Feed: The script uses the OpenCV library to capture the video feed from the webcam. It then passes each frame through the 'predict_class' function to classify the fruit type.

6. Displaying Output: The predicted fruit type is displayed on the screen using the OpenCV library.

7. Exiting the Program: The script waits for the user to press the 'q' key to exit the program.

In summary, this script uses a pre-trained CNN model to classify fruit types from a live video feed and displays the predicted output on the screen. The script also uses the gtts library to convert the predicted output into speech and plays it using the playsound library.

## VI. RESULTS

This work, fully automatic and grading of multiple fruits are proposed. In this project, images of different fruits are given as Input via a live video stream and fruit image is classified live , their various feature are extracted based on which we grade them as Good and Bad. The system also intent to predict the ripeness of fruits purchased and evaluate the quality of the fruits more precisely. Sensors are used to check moisture content in fruits and gas released from fruits and other food items. The system attain better classification and fruit detection with maximum accuracy and enhance the production yield.

## VII. CONCLUSION

This research paper has presented a novel approach to preventing food poisoning by using deep learning algorithms to detect the quality of fruits. The results show that the proposed system can accurately identify the quality of fruits and provide a warning if the fruit is potentially hazardous for consumption. This system can be integrated into the food industry to ensure the safety of fruits and prevent food poisoning outbreaks.

Furthermore, the proposed system has the potential to reduce food waste and increase the efficiency of the supply chain by identifying fruits that are not suitable for consumption at an early stage. This can lead to significant cost savings for the industry and contribute to sustainable food production practices.

Overall, this research provides a promising solution for preventing food poisoning and improving food safety using modern deep learning techniques. Future work can expand this system to other types of fruits and explore its potential applications in other food industries.