

Project Report on
Fruit Ripeness Detection using Computer Vision

Submitted to
Amity University Uttar Pradesh



In partial fulfilment of the requirements for the award of the degree
of

Bachelor of Technology

in

Computer Science and Engineering

by

Shaurya Guliani

A2305219086

Under the guidance of

Dr. (Ms) Vasudha Vashisht

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

AMITY UNIVERSITY UTTAR PRADESH

DECLARATION

I, Shaurya Guliani, student of B.Tech (5CSE-2X) hereby declare that the project titled “**Fruit Ripeness Detection using Computer Vision**” which is submitted by me to Department of Computer Science and Engineering, **Amity School of Engineering Technology**, Amity University Uttar Pradesh, Noida, in partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering, has not been previously formed the basis for the award of any degree, diploma or other similar title or recognition.

The Author attests that permission has been obtained for the use of any copyrighted material appearing in the Dissertation / Project report other than brief excerpts requiring only proper acknowledgement in scholarly writing and all such use is acknowledged.



Shaurya Guliani

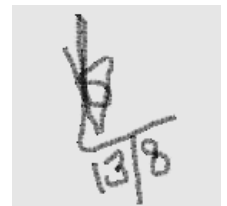
Date:30/06/2021

A2305219999

5CSE-2X (2019-23)

CERTIFICATE

On the basis of the declaration submitted by Shaurya Guliani (Enrolment No.: A2305219086) ,student of B.Tech Computer Science and Engineering, I hereby certify that the project report entitled **“Fruit Ripeness Detection using Computer Vision ”**which is submitted to the Department of Computer Science, Amity School of Engineering and Technology, Amity University, Noida, Uttar Pradesh in partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering is an original contribution with existing knowledge and faithful record of work carried out by him under my guidance and supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.



Dr. Vasudha Vashisht
(Associate Professor)

Department of Computer Science

Amity School of Engineering and Technology

Amity University, Uttar Pradesh

Date: 30/06/2021

ACKNOWLEDGEMENT

The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. I would like to thank **Prof. (Dr.) Abhay Bansal**, Head of Department-CSE, and Amity University for giving me the opportunity to undertake this project. I would like to thank my faculty guide **Dr. (Ms) Vasudha Vashisht** who is the biggest driving force behind my successful completion of the project. She has been always there to solve any query of mine and also guided me in the right direction regarding the project. Without her help and inspiration, I would not have been able to complete the project. Also I would like to thank my batch mates who guided me, helped me and gave ideas and motivation at each step.



Shaurya Guliani

ABSTRACT

This report illustrates the detailed functioning and design of a “**Fruit Ripeness Detection Application**” programmed using **Python** which uses a trained categorical **Convolutional Neural Network (CNN) Model** to scan an uploaded image of a fruit for characteristics and predict whether the fruit is unripe, ripe or overripe.

The application is a **Web Based App** which asks the operators to browse and upload any fruit image from the system and scans the obtained features through the created database for similarities and returns the predicted ripeness of the fruit along with its name.

Keras and Tensorflow libraries are used along with Python for developing the CNN Model and preparing it for its purpose. The designing of the created CNN is done using two sets of subsequent two initial layers including the Convolutional Layer, which is the first layer of the architecture used to extract characteristics from the object image, and the Pooling Layer, which converts the obtained image output to a lower dimensioned version. **Flask** framework is used for deploying the application on a local host and linking the created model with it. **HTML (Hypertext Markup Language)** along with **CSS (Cascading Style Sheets)** are used for creating and styling the application.

The above mentioned application, if integrated with a robotic AI, can prove efficient for the agriculture sector by increasing manpower and performing sorting functions which are time consuming.

LIST OF FIGURES

Figure 1.1: Different Ripeness stages of a Banana

Figure 2.1: High-level detail withdrawal and identification

Figure 2.1: Use of rediripe stickers on ripe apples and a pear showing blue color as the fruits are perfectly ripe.

Figure 2.2: Microsoft Visual Studio Code

Figure 2.3: Python

Figure 2.4: Tensorflow and Keras

Figure 2.5: Structure of a Convolutional Neural Network

Figure 2.6: Flask

Figure 3.1: Index.html webpage

Figure 3.2: Template.html webpage

Figure 3.3: Deployment of application on local host

Figure 3.4: Home page of the created Fruit Ripeness Detection Application

Figure 4.1: Browsing and uploading of input to the app

Figure 4.2: The application detecting an image of unripe oranges as unripe bananas due to the similarity in the color scheme- A limitation of the Fruit Ripeness Detection application

Figure 4.3: The application detecting the photo of a table as an unripe banana- A limitation of the Fruit Ripeness Detection application

Figure 4.4: Testing for unripe banana

Figure 4.5: Testing for ripe banana

Figure 4.6: Testing for overripe banana

Figure 4.7: Testing for unripe orange

Figure 4.8: Testing for ripe orange

Figure 4.9: Testing for overripe orange

Figure 5.1: Root-Ai's Tomato Ripeness Detecting robot capable of detecting ripeness and collect tomatoes non-destructively

TABLE OF CONTENTS

<u>DESCRIPTION</u>	<u>PAGE NO.</u>
Declaration.....	2
Certificate.....	3
Acknowledgement.....	4
Abstract.....	5
List of Figures.....	6
Table of Contents.....	7
Chapter 1: Introduction.....	8
Chapter 2: Literature Review.....	11
2.1: Microsoft Visual Studio Code.....	13
2.2: Python.....	13
2.3: Tensorflow.....	14
2.4: Keras.....	14
2.5: Convolutional Neural Networks.....	15
2.6: Flask.....	16
Chapter 3: Methodology.....	17
3.1: Model Construction.....	17
3.2: Creation of Web Application.....	18
3.2.1: Index.html.....	18
3.2.2: Template.html.....	19
3.3: Linking the Model to the App.....	19
Chapter 4: Working and Testing.....	21
Chapter 5: Future Scope.....	26
Chapter 6: Conclusion.....	27
Chapter 7: References.....	28

INTRODUCTION

A Fruit's Ripeness is a term adopted for signify a fruit's quality. Perfectly ripe fruits tends to be perfectly tender in softness and rich in taste than the other stages, largely unripe and overripe.

In a world where food is an important necessity and where a good chunk of the World's Economic Production (6.4%) is provided by the agriculture sector, it becomes an important responsibility for agricultural producers to provide the best outcomes.

Largely, the lifetime of a fruit's growth is divided into 3 stages:

Unripe: The Initial stage of the fruit in which the fruit is theoretically not considered ready to eat as it lacks the appropriate texture, taste and softness required. In most of the fruits, this stage is easily identified with the presence of green color and reduced size of the fruit.

Ripe: At this stage, the fruit is plucked. The natural color of the fruit is observed and the fruit is also mature enough. Best taste of the fruit is provided at this state.

Overripe: When a ripe fruit is kept for a significantly long time, it gets exposed to the microbes in the atmosphere because of which the fruit gets more soft and the taste becomes less satisfactory. This stage is identified by a major part of the fruit's outer skin becoming black in most of the cases.



Fig 1.1: Different Ripeness stages of a Banana

The fruit classification into ripe, unripe and overripe is possible by a person by looking at the fruit's physical appearance.

This task of classification of fruits on the basis of their ripeness can be automated with the help of programming languages like Python along with machine learning libraries and Computer Vision thereby making the task easier and less time consuming.

This project can help in the advancement of technologies used in the agricultural sectors generating possibilities to look at machine learning differently.

With the increasing use of language programming to extract high level features from images, more projects and researches using technology for accurate extraction of characteristics and features. All this became possible due to the introduction of Computer Vision to the technological world.

Computer Vision impersonates a human's life system to carry out processing and decision making as a human brain. This is achieved by training a database model which, on the basis of predefined database of features, could predict the ripeness of a fruit solely by looking at it. This will decrease the time consumption of this task and will also help in increasing the workforce. A database of photos stored in different categorical folders will be used to create a CNN (Convolutional Neural Network) Model which could perform the above defined task.

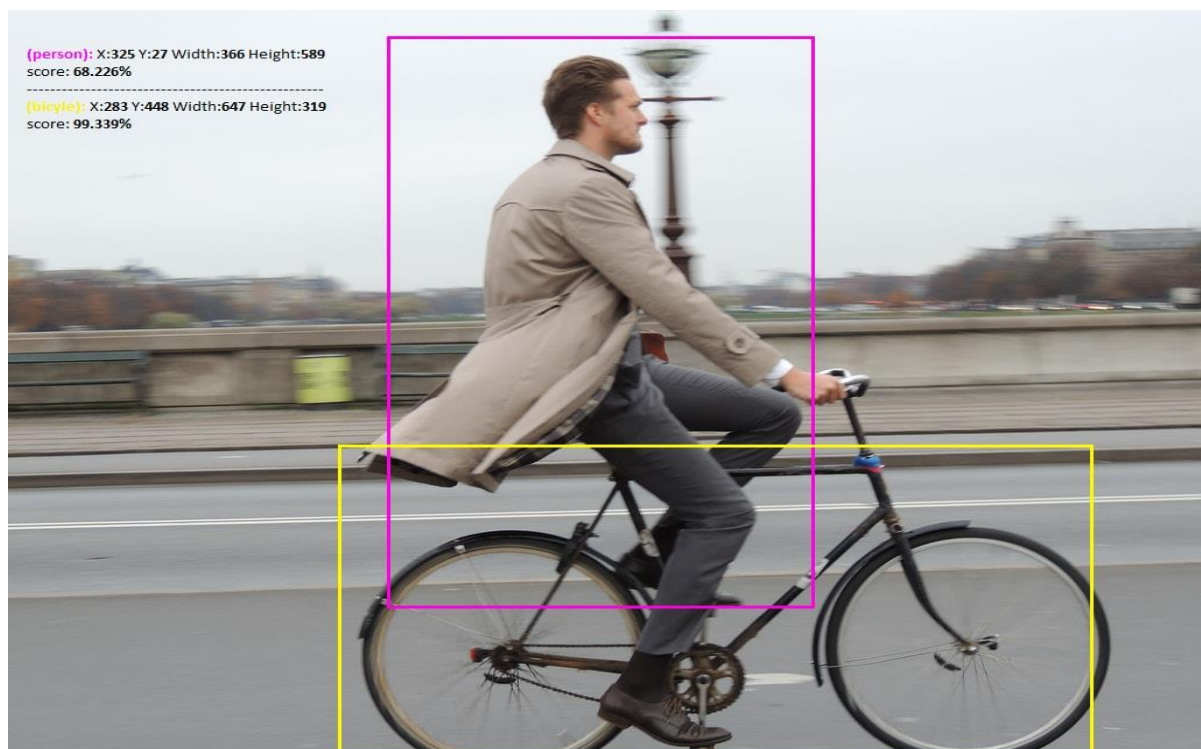


Fig 2.1: High-level detail withdrawal and identification

This project report will depict the different aspects and entities used for the creation of a fully functional CNN architecture to identify a fruit's ripeness by looking at its image and will present an explanation of the architecture.

The report is organized into 4 sections:

- I. First section “Literature Review “gives a brief history about all the previous work done on “fruit’s ripeness detection” with an introduction about Computer Vision.
- II. Second section “Methodology” will contain information and functioning of all the technologies and ideas used during the creation of the project.
- III. Third section “Working and Testing” talks about the implementation and results of the created application.
- IV. Fourth section “Future Scope” will consist of all the “Future Ideas and possibilities” that can make this project more important and valuable.

LITERATURE REVIEW

The first experiment of detecting a fruit's ripeness using technological means dates back to 2006 when a mere sticker was used to detect the readiness of an apple in Tucson, Arizona created by **Mark Riley**, a scientist of the **University of Arizona** during that year. The stickers were call the “rediripe stickers” [1]

It used the produced quantity of ethylene, a gas excreted after the fruit has ripened. The sticker was synthesized in a way to turn itself from yellow to blue as the amount of produced ethylene is increased. [2] Though the experiment was successful, there were other complications to be considered. The ethylene production of a fruit varies largely with the variation in weather and hence, the use of chemical analysis in determining the fruit's ripeness gave some inaccuracies.



Fig 2.1: Use of rediripe stickers on ripe apples and a pear showing blue color as the fruits are perfectly ripe.

Some modifications can be on the chemical methods of ripeness detection to make it more accurate but still, cannot fix the high time consumption error of the method. Hence after the popularization of artificial intelligence and more importantly computer vision, people started taking a keen interest in automating the process. Researchers considered creating a machine learning model and to retrieve maximum characteristics to do categorization. [3]

Computer Vision which deals with ways for computers to access high-level information from photographic images or digital videos. The first idea of Computer Vision dates back to late 1960s where it first started in Universities which were exploring Artificial Intelligence. Its primary aim was to impersonate visual system of the human eye to move a step closer to the development of intellectual functioning of robots. Foundations of many of the existing frameworks like photo-edge extraction and line labelling in modern Computer Vision are a result of studies made in the 1970s.[4]

The first instance of machine learning can be dated back to 1952 which saw the first computer which was able learn while running. Arthur Samuel created a game which simulated checkers.

1958 saw the first creation of an artificial neural network by Frank Rosenblatt which was used in recognition of different patterns and shapes.

Since then, machine learning became an important and interesting aspect for researchers and tech enthusiasts as the concept of replicating the attributes of a living human being fascinated them.

The project is created using **Microsoft Visual Studio Code** in **Python** language. It includes the use of **Tensorflow** and **Keras** libraries to create a **Convolutional Neural Network (CNN) Model** using a huge dataset of images including those of different fruits in different stages- unripe, ripe and overripe. The CNN model serves as the comparison structure to test image and predict the ripeness.

The CNN model is deployed on local host using **Flask** library, a Python framework used for creating web applications and the **Numpy** library, which changes the image data to a linear array which is easily read and predicted by the model.

The creation and styling of the application is done using **HTML and CSS**.

The versions of each library used are mentioned below:

- **Microsoft Visual Studio Code** – 1.57.0
- **Python** – Python 3.9.1
- **Tensorflow** – Tensorflow 2.5.0
- **Flask** – Flask 2.0.0
- **Numpy** – Numpy 1.19.5

Microsoft Visual Studio Code

Initially announced on 29 April 2015, Visual Studio Code is a source-code editor created by Microsoft having a vast installable extensions supporting numerous languages including Java, Python, C++ etc.[5][6]

Having its own terminal and multiple directory environment where, Visual Studio Code, allows the user to use different languages at one place thereby helping in structuring the project. [7]

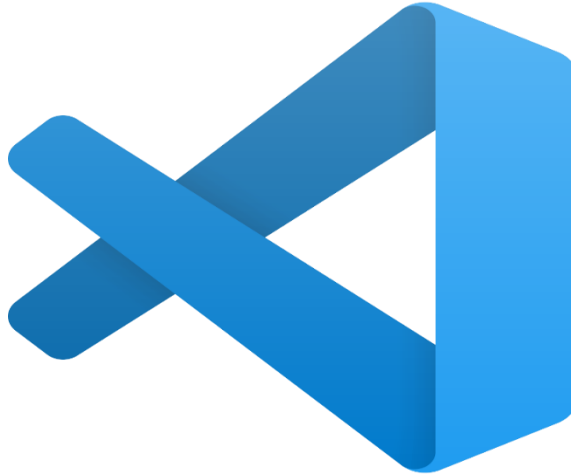


Fig 2.2: Microsoft Visual Studio Code

Python

Python is a high level language which uses an object oriented approach thereby, allowing coders to write efficient programs. Formulated in the end of 1980s, its execution started in 1989. Having an English like syntax, it provides simplicity and high understanding to the written code making it easier to interpret. Inclusion of many types of external libraries make it the best choice for projects.

Its support of machine learning libraries Tensorflow and keras making it efficient in creation of artificial intelligence projects.



Fig 2.3: Python

Tensorflow

Tensorflow, a library based on differential programming and dataflow created and was released by Google in 2015, is a platform which helps the user in deploying and creating of machine learning models.[8]

Tensorflow allows the user to choose from numerous abstraction levels. Models are created and trained using Keras API which helps to start learning with tensorflow and ML fun,easy and enjoyable.

Mainly developed using C++,Python and CUDA (GPU programming language for Nvidia), tensorflow is supported as an external library with Java, JavaScript, C#, Matlab etc. along with Python and C++.

Keras

Keras is a tensorflow library supported by python which provides modules and functions for artificial neural networks. Before version 2.4, Keras supported many backends including tensorflow but following the release of tensorflow 2.4.0, Keras only supports tensorflow as backend. [9]

It is a quite user friendly extensible module which the supports the user in experimenting with deep neural networks. It consists of function which are neural-network building blocks like optimizers, layers and tools allowing to work with images. Keras consists support for Convolutional as well as Recurrent Neural-Networks.



Fig 2.4: Tensorflow and Keras

Convolutional Neural Networks

Convolutional Neural Networks is a sub-category of Deep Neural Networks with are used to analyze and categorize images on the basis of their visual features. CNNs are based on weight sharing architectures consisting of layers. [10] A basic Convolutional Neural Network consists of 3 basic layers:

- **Convolutional Layer:** Used to change the input data format into a vector by applying a Convolutional operation.
- **Pooling Layer:** It implements a filter to the obtained vector for reducing the dimensions of it therefore, decreasing the number of features the model has to learn.
- **Fully Connected:** It segments the model in a number of layers in such a way that each vertex of a layer is connected to each of the subsequent one. This helps in compiling all the obtained data together and returning a highly accurate result.

A CNN can be thought of as a database of visual imagery features of different classes of object which can distinguish the images into the available classes.

When an image is fed into the trained model, the layer work on creating activation maps to extract features. These features are used in the subsequent layers for allotting the most accurate class to the input object image.

The detail withdrawal of the input is done with the convolutional and pooling layers and classification is henceforth, handled by the fully connect layers.

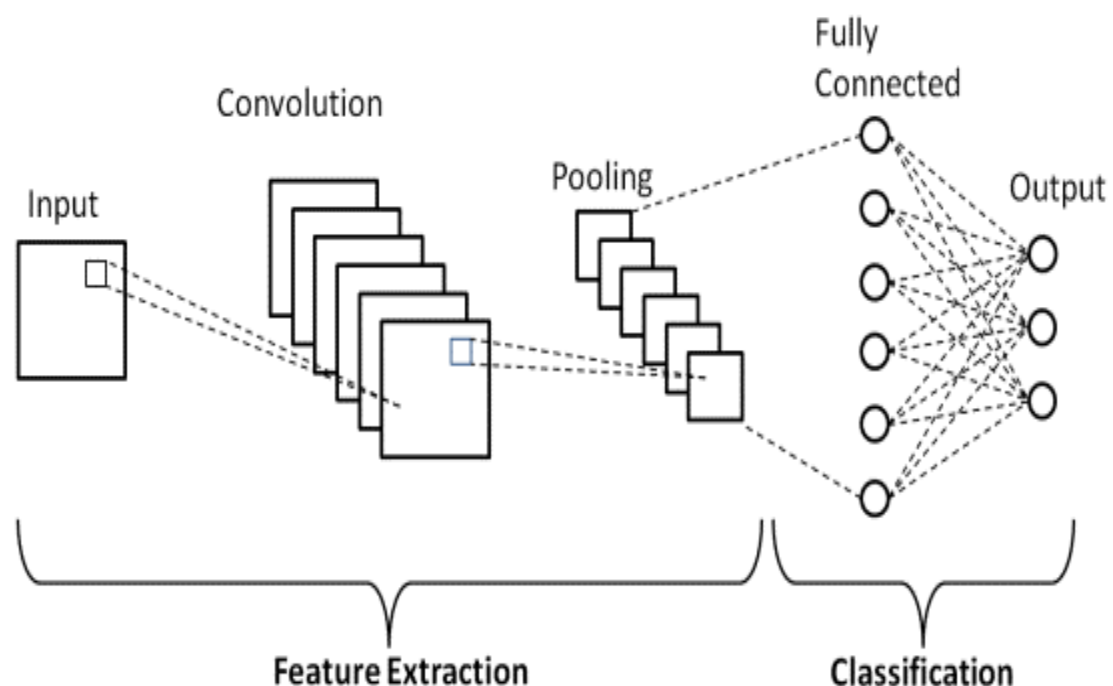


Fig 2.5: Structure of a Convolutional Neural Network

Flask

Flask, created by Armin Ronacher , is a web micro framework created in Python which is used for creating web applications. [11] Flask also links a Machine Learning model to an app and give it a runnable interface. It works with a webpage creating language and deploy the ML model on a local host.



Fig 2.6: Flask

METHODOLOGY

The whole project is divided into 3 parts:

- **Creating the CNN model for prediction purposes**
- **Constructing and Styling the Application with HTML and CSS**
- **Linking the CNN model with application and deploying it on Local host using Flask**

Model Construction

For the model, a huge database of fruit images at all ripeness stages is required which will be used for the creation model.

The database has two sub categories namely **Train** and **Test**. “Train” contains the images which will train the CNN model. These images will undergo numerous processes including shearing, rescaling, rotation etc. for the withdrawing of details. “Test” will store images which will be used model numerous time to make the model accurate.

The images in **Train** and **Test** are then divided into 6 folders: **Unripe Bananas, Ripe Bananas, Overripe Bananas, Unripe Oranges, Ripe oranges, Overripe Oranges.**

Image distribution of each folder is given below:

Train:

Unripe Bananas- 103

Ripe Bananas- 175

Overripe bananas- 245

Unripe Oranges- 100

Ripe Oranges- 165

Overripe Oranges- 178

Test:

Unripe Bananas- 31

Ripe Bananas- 43

Overripe bananas- 61

Unripe Oranges- 30

Ripe Oranges- 41

Overripe Oranges- 44

After collecting images and creating the database, the CNN model is created using Python and Keras. A CNN is a layered structure and hence many layers are used in creation of the model:

1. First layer is Convolutional layer which is added using the “Conv2D” function.
2. The second layer is the pooling layer which consists of a filter matrix used for dimension reduction. Max pooling is used in this layer with pool size as **2**.
3. The third and the fourth again consists of another convolutional and pooling layer having same specifications. (The model consists of 2 sets of Convolutional and Pooling layers)

4. The fifth layer is the **Flatten** function which transforms the obtained vectors to a Flat **1D** array.
5. Finally, a **6 layered** full connection is done using the obtained flat arrays to create an Artificial Neural Network and hence, a CNN.

The CNN is finally compiled using **adam** optimizer which comes with Keras.

A total of 12 batches are used while the training and testing of the CNN and epochs size is kept as 20 while the model compilation.

Creation of Web Application

The Web Application consists of two pages:

1. Index.html - to upload the image to detect its ripeness.
2. Template.html – displaying the uploaded image with its predicted ripeness

Index.html:

The Index.html webpage is the homepage which is visible after opening the application. It asks the user to choose a jpg/png image. It consists of two buttons:

1. **Choose files** button which will direct the user to browse any jpg/png file present in the system and upload it.
2. **Detect** button which will use the uploaded image and run it through the CNN model for prediction of the ripeness.

Fruit Ripeness Detector

**Upload Image to Predict
the Fruit's Ripeness**

Upload Image:

No file chosen

Fig 3.1: Index.html webpage

Template.html:

The Template.html page is the page which will be displayed after the ripeness of the uploaded image's fruit is detected. It displays the selected image along with its predicted ripeness. The page consists of a single button which asks the user to select another fruit image for the detection of its ripeness.

Detecting the Fruit's Ripeness

Selected Image:



{{text}}

Choose another image

Fig 3.2: Template.html webpage

Linking the Model to the App

Flask creates a standard application which can be edited and programmed using created html pages and imported functions and libraries:

```
app = Flask(__name__)
```

Classes are mentioned which will be displayed after the detection is done:

```
classes = ['Ripe Banana', 'Ripe Orange', 'Overripe/Rotten Banana', 'Overripe/Rotten Orange', 'Unripe Banana', 'Unripe Orange']
```

Once, after creation of the basic application, functions can be defined for linking created webpages and navigating the user interface.

Two functions, index() and upload() are defined.:

Index() function links the index.html webpage to the main application as its homepage when it is booted

Upload() saves the uploaded image in “images” folder in the application directory.

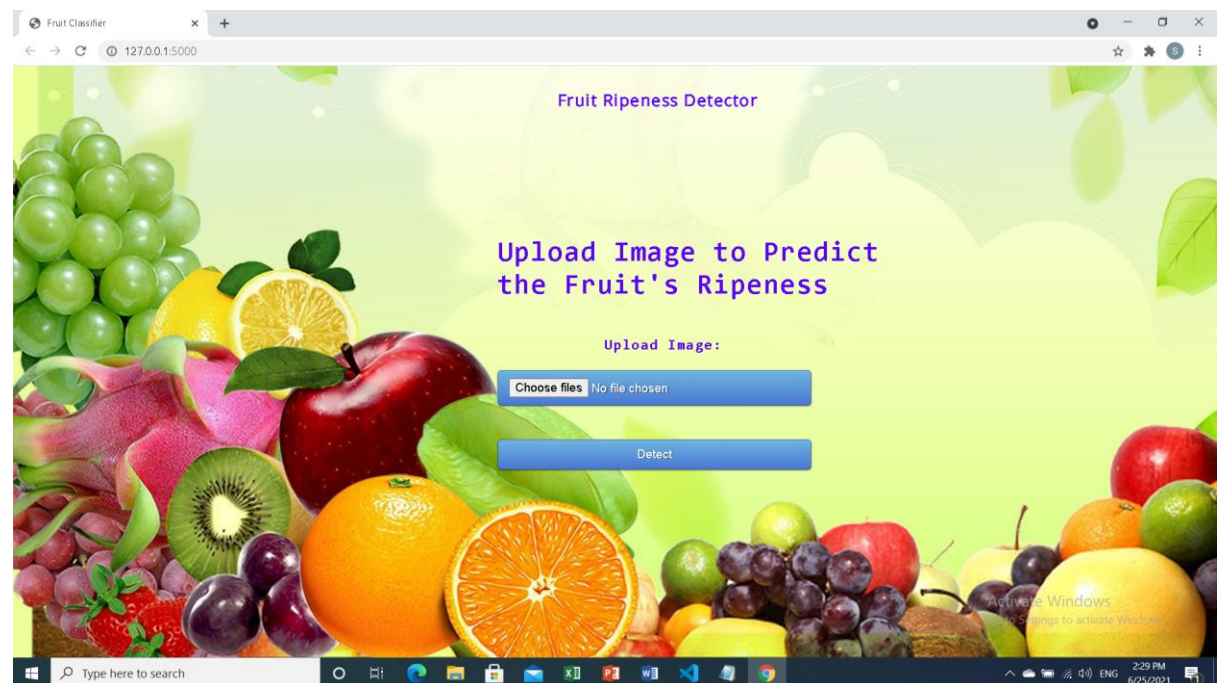
Load_Model() inbuilt function is used for loading the created CNN model to the app. After the loading is done, the latest uploaded file in “images” folder is used as a test image and it’s ripeness is predicted using the predict() function.

The template.html webpage is displayed with the current used photo uploaded into it and the calculated predicted ripeness.

The created application is deployed on the local host after the application code is run:

```
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Fig 3.3: Deployment of application on local host



*Fig 3.4: Home page of the created **Fruit Ripeness Detection** Application*

WORKING AND TESTING

The created Fruit Ripeness Detection application, on booting, allows the user to choose a jpg/png image of a fruit and upload it to predict its ripeness.

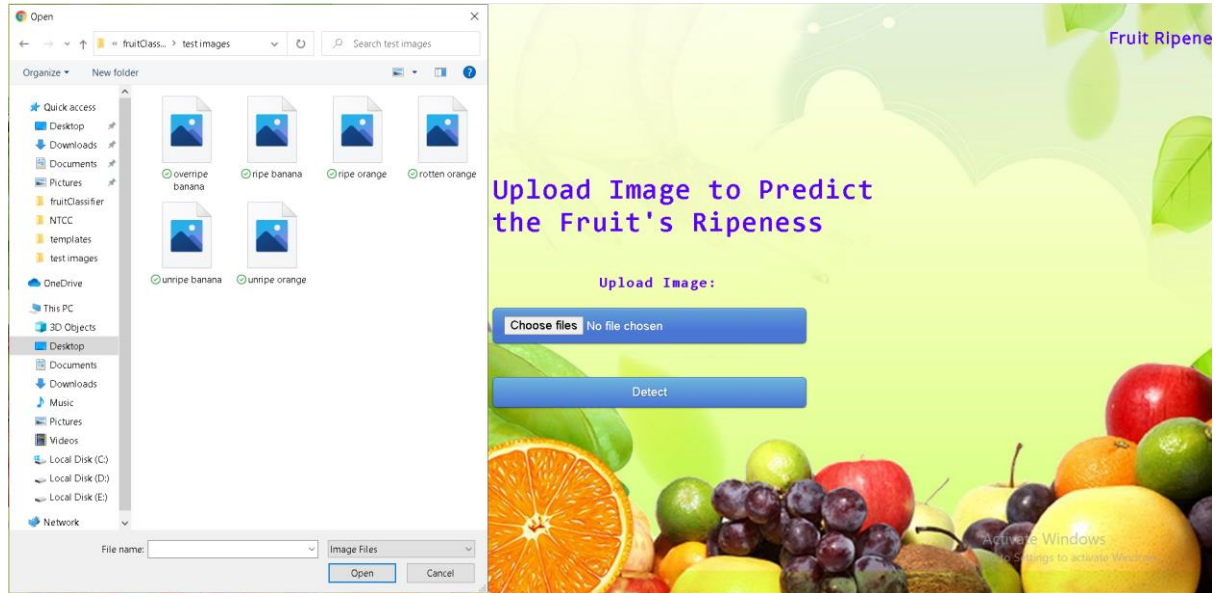


Fig 4.1: Browsing and uploading of input to the app

Once the image is uploaded, the application saves the uploaded image in the “**images**” folder in the main directory. The saved image is converted into matrix to make sure it is easily understandable by the model.



The obtained matrix now undergoes the same processes as the dataset images went through allowing the application to easily compare and categorize the uploaded image into the available categories.

After predicting and categorizing the browsed image to its appropriate category, the application allows the operator to browse another image.

The CNN model is trained in 20 passes (epochs) in which:

- The highest accuracy of 0.9482 was achieved in the 19th pass

```
Epoch 19/20
81/81 [=====] - 9s 106ms/step - loss: 0.1460 - accuracy: 0.9482 - val_loss: 0.1852 - val_accuracy: 0.9440
```

- The lowest accuracy of 0.3944 was achieved in the 1st pass

```
Epoch 1/20
81/81 [=====] - 41s 473ms/step - loss: 1.4239 - accuracy: 0.3944 - val_loss: 0.7853 - val_accuracy: 0.6880
```

Even though, the application works smoothly and is highly accurate in serving its purpose, it has some cons:

1. The trained CNN model focuses on the color scheme of the fruits a little more than their shape and visual texture which might lead to difficulties and inaccuracy in the fruit's recognition. The application will surely detect the fruit's ripeness but have a very little chance of failing to recognize the fruit it is predicting.

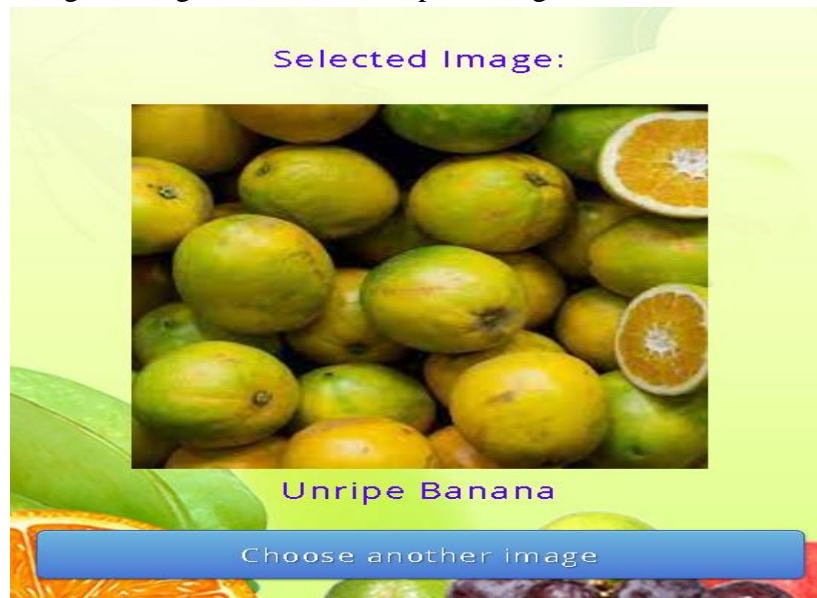


Fig 4.2: The application detecting an image of unripe oranges as unripe bananas due to the similarity in the color scheme- A limitation of the Fruit Ripeness Detection application

2. The model is trained using 6 different databases having multiple images of that category to increase the system's accuracy but might find it difficult to categorize an object which are unrecognizable to the model such anything other than bananas and oranges. This will lead to the system categorizing the object mistakenly to the most suitable one. Hence, it is advisable to only upload images of bananas and oranges for the ripeness detection process.



Fig 4.3: The application detecting the photo of a table as an unripe banana- A limitation of the Fruit Ripeness Detection application

The application is trained to function for two fruit – bananas and oranges – and classify them further as ripe, unripe or overripe. Testing for all of these 6 categories were performed using the application and accurate results were produced:

Unripe Banana:

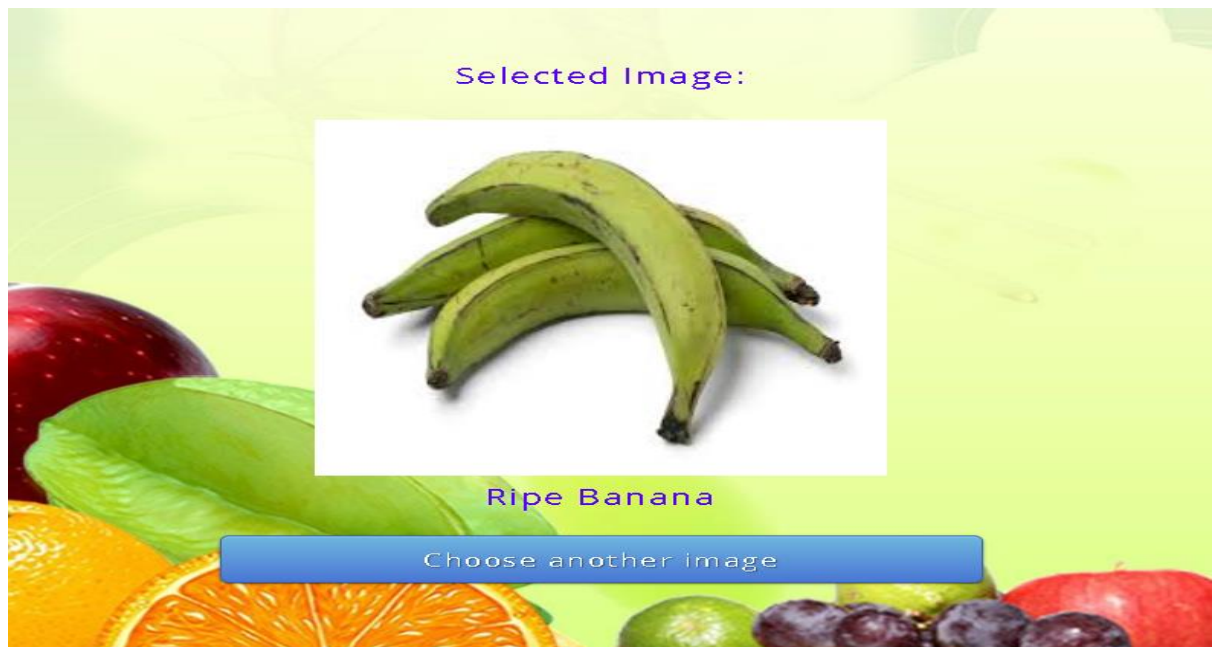


Fig 4.4: Testing for unripe banana

Ripe Banana:



Fig 4.5: Testing for ripe banana

Overripe Banana:



Fig 4.6: Testing for overripe banana

Unripe Orange:

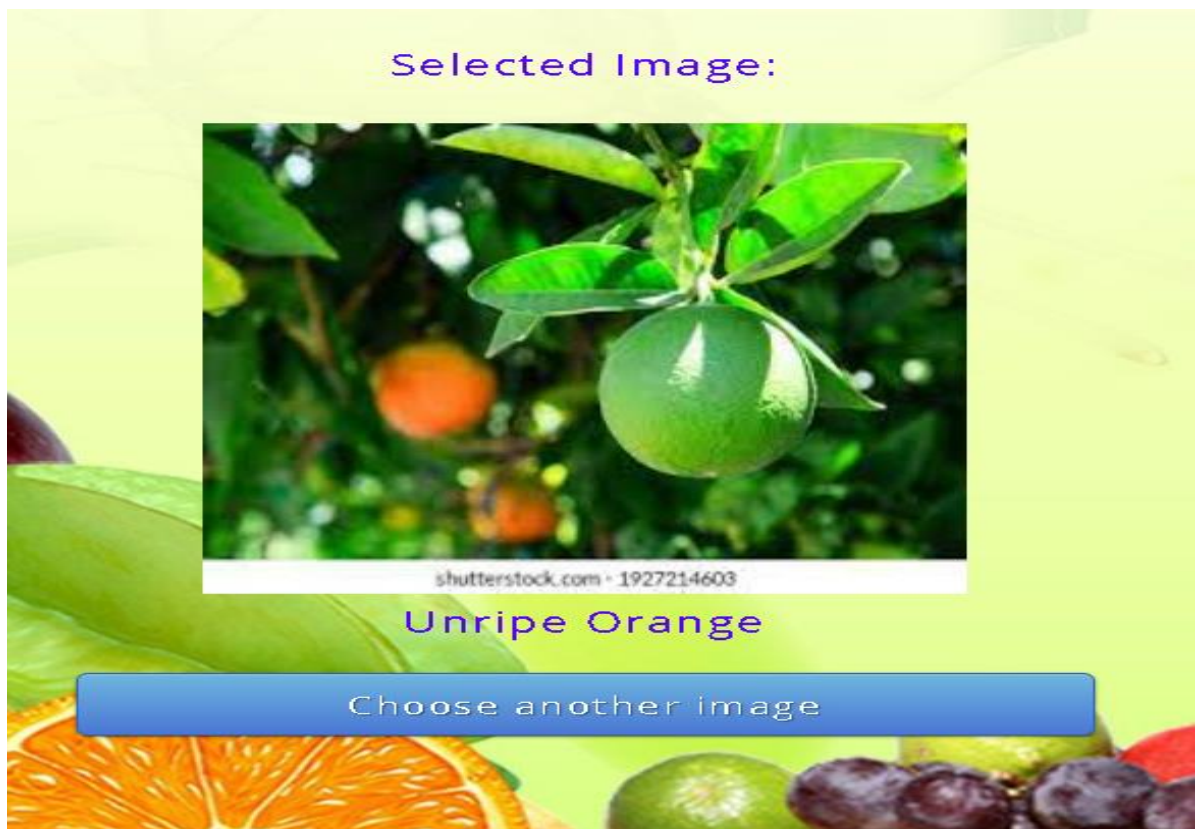


Fig 4.7: Testing for unripe orange

Ripe Orange:



Fig 4.8: Testing for ripe orange

Overripe Orange:



Fig 4.9: Testing for overripe orange

FUTURE SCOPE

The introduction of automatic visual imagery based Fruit Ripeness Detection applications and systems will help in changing the agriculture sector completely for good. These systems can handle the time consuming work of classification and handpicking of fruits easily and fasten the process which takes a big load of workforce. The application can be integrated with artificial humanoids to give them a functional state for the purpose.



Fig 5.1: Root-Ai's Tomato Ripeness Detecting robot capable of detecting ripeness and collect tomatoes non-destructively

The above displayed bot was created by “Root AI” – a Somerville, Massachusetts based start-up establishment – in May 2019 can easily detect perfectly ripe tomatoes and pluck them for further processing. Similar to this, numerous models can be used to automate the agriculture sector.

Besides the processing stage, the identification process can help during the distributing stage allowing people to buy agricultural produces from automatic machines.

The future for Computer Vision is quite well in agriculture opening up new ideas and possibilities aiding the producing and consuming population alike.

CONCLUSION

The project report illustrated the design and functioning of the “Fruit Ripeness Detection” application which works on bananas and oranges to predict their ripeness and classify them as – unripe, ripe and overripe.

The application is programmed using Python along with Tensorflow, a library for creating ML projects.

The methodology presented the creation of a Convolutional Neural Network Model using a huge database of image to create 6 classification categories and use it for the prediction of a fruit’s ripeness. It used the Keras sub-library of Tensorflow which mainly helps in creation of CNN models and training them.

The created CNN model consisted of a total of 6 layers including 2 sets of Convolutional and Pooling layers, a flattening layer and a complete connection layer. Max pooling is the technique used in the implementation of pooling layer in the CNN.

After the creation of the model, the CNN was trained in 20 passes where it attained the highest accuracy of 94.82% making it quite accurate to serve the purpose.

A complete application was designed and styled using HTML and CSS and Flask deployed the created model on local host with the application.

Testing for all the 6 categories was done using different images of bananas and oranges and the application proved to be quite accurate during the ripeness detection process making the project a success.

With the new upcoming technologies and more sectors getting automated, Fruit classification and ripeness detection applications can prove to be efficient and unique for the agricultural sector stepping towards new and advanced world with modern technologies in every sector possible

REFERENCES

- [1] "Professor invents 'ripeness' sticker". USA Today. July 27, 2006. Archived from the original on October 13, 2012
- [2] McGinley, Susan. "Inventing the RediRipe® Sticker." (2007).
- [3] Puttemans, Steven & Vanbrabant, Yasmin & Tits, Laurent & Goedemé, Toon. (2016). Automated visual fruit detection for harvest estimation and robotic harvesting. 10.1109/IPTA.2016.7820996.
- [4] Richard Szeliski (30 September 2010). Computer Vision: Algorithms and Applications. Springer Science & Business Media. pp. 10–16. ISBN 978-1-84882-935-0.
- [5] Montgomery, John (April 29, 2015). "BUILD 2015 News: Visual Studio Code, Visual Studio 2015 RC, Team Foundation Server 2015 RC, Visual Studio 2013 Update 5"
- [6] "Monaco Editor". microsoft.github.io/monaco-editor.
- [7] "Language Support in Visual Studio Code". Visual Studio Code. October 10, 2016. Retrieved 2016-10-12
- [8] Metz, Cade (November 9, 2015). "Google Just Open Sourced TensorFlow, Its Artificial Intelligence Engine". Wired. Retrieved November 10, 2015.
- [9] "Release 2.4.0". 17 June 2020. Retrieved 18 June 2020.
- [10] Zhang, Wei (1988). "Shift-invariant pattern recognition neural network and its optical architecture". Proceedings of Annual Conference of the Japan Society of Applied Physics.
- [11] "Flask Foreword". Archived from the original on 2017-11-17.