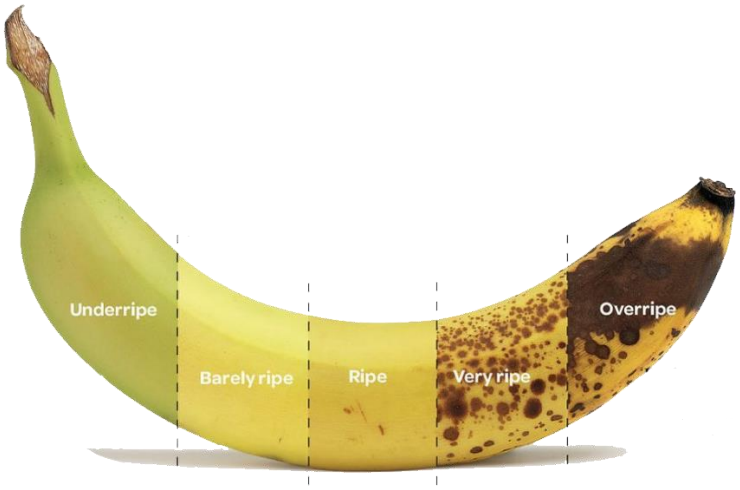
המכללה האקדמית להנדסה בכרמיאל
בראודה

Capstone Project Phase B

Fruit image Classification and identification of its ripeness
Using Deep Learning

Final Project in Software Engineering (Course 61999)



Project code: 23-1-R-7

Shani Froik, mail: shanifroik@gmail.com

Lior Saghi, mail: liorsaghi@gmail.com

Supervisor: Mr. Ilya Zeldner, mail: zeldner26@gmail.com

Content

Abstract

In recent years Fruit classification has become a common subject in the computer science field. Fruits can be identified by an image using data- it extracts features that are processed in deep learning and gives information based on the image.

In this project we focus on classifying fruits with an image, especially the type of fruit and its ripeness, using deep learning. The first step is to classify the type of fruit by using the Visual Geometry Group 16-layer network (VGG16) which we found would yield the best results for our project. We used to classify on a general database of different types of fruit. Once we know the type of fruit, the second step is to understand the level of ripeness of the fruit. We took the banana fruit to check for ripeness: we used VGG16 on a database of the ripeness of the bananas by days and thus we can give an estimate of how many more days the banana can be eaten and when it is rotten and it is better not to eat it. Similarly, it will be possible to perform this on any other fruit and thus predict the ripeness of each fruit.

There are many uses to this project. It can help common people choose the finest fruit at the supermarket. It can also prevent people from eating unripe fruit. Another use is to help disabled populations with vision impairments and even blindness giving them a way to overcome their disability.

Keywords:

image classification, fruit classification, fruit ripeness, deep learning, neural networks, VGG16, Visual Geometry Group.

1. Introduction

Eating fruit is a recommended way to improve everyone's health and in the long run reduce the risk of disease. Fruits are a good source of minerals, energy, and vitamins and dietary fiber. There are many types of fruits such as: apple, orange, banana, pear, avocado, watermelon, mango and many more. Fruit classification plays an important role in many applications - whether industrial such as factories, agriculture or services such as supermarkets - whether it helped the common man to choose ripe fruit and a person with a disability to identify the fruit he intended to purchase or whether it helped the cashier to automatically select the right type of fruit and determine Priced accordingly.

Determining the condition of the fruit, whether ripe, unripe or scaled, is often performed by a manual sorting method. This method has been around for years, but this method is time-consuming and always involves human intervention. For example, in agriculture, a lot of labor is required to classify the fruits, and despite this, the sorting manual does not give good results in many cases. The same is the case in many other fields and again also for the common man who wants to eat a fruit that is already ripe but has not started the process of rotting.

Therefore, technological techniques are required in order to classify which fruit it is and then determine its condition. In this project, a machine learning-based approach is presented for the classification and identification of fruit by using an image and, after the classification, the assessment of the ripeness of the fruit.

At the beginning of the research, we thought that the classification process would be carried out in several stages when first we will enhance the image by Fuzzy Type-II, then Convolutional Neural Network (CNN) will extract its convolution features from the image - it will detect the edges, the color features and then the texture and shape features of the fruit. After that, Recurrent Neural Network (RNN) will be used to label optimal features and Long Short-Term Memory (LSTM) will classify the images using the optimal features that CNN and RNN extracted from the image with the help of deep learning and then, the ripeness of the fruit will be checked by extracting features from the fruit image such as the RGB color space, its boundaries in the image in order to insert this data into a decision tree that will classify the fruit according to its ripeness.

After in-depth research and after reading the articles [24] and [25], we discovered that it is possible to classify both the type of fruit and the ripeness of the fruit using only a deep learning model which gives higher accuracy compared to the use of a decision tree to classify the ripeness of the fruit. In addition, we investigated which model would give us the highest accuracy among the following models: VGG16, ResNet50, MobileNet, InceptionV3, Desnet121, InceptionResNetV2, Xception and CNN. We found that according to the articles we should use VGG16 because it seems to give better accuracy compared to the other models.

After we were able to classify the type of fruit we wanted to predict how long the fruit had left to ripen before it was rotten. Therefore, we chose to monitor the ripeness of a banana, we built a dataset by days and with the help of the VGG16 model we were able to predict how many days a banana has left until it is considered rotten and how many days a banana will be ripe. This prediction shows that for any fruit under certain conditions and at a certain temperature it is possible to predict the number of days left for the fruit to be edible by using a VGG16 deep learning model.

2. Background and Related Work

2.1 Neural Network

Neural network is a mathematical method that teaches itself to process data. The method is inspired by the cognitive function that occurs in the human brain and simulates a network of neurons. It is a network that acts as a response to certain stimuli so that they take in information, process it and respond. A neural network deals with building virtual models of the nerve cell layers that contain several information units that are linked together and represent input and output. The input is the received information and passes through hidden units that process it, and then an output comes out. Each such processing basically performs a simple mathematical operation, but when there are connections between them, they can perform complex operations.

2.2 Convolutional Neural Network (CNN)

Convolutional Neural Network is a type of neural network that is primarily used to receive image input and process it.

A convolutional network often uses convolution layers, pooling layers and fully connected layers.

2.2.1 Convolution layer

Receives as input a tensor (multilinear function) representing an image. Outputs as a feature map - a tensor with a more abstract image.

This is done cyclically and each tensor input comes out as an output for the input of the next convolution layer.

2.2.2 Pooling layer

Receives a tensor as input and outputs a smaller tensor.

In the pooling layer, a lot of information is lost, but this layer reduces the computational complexity and thus saves processing time.

There are many types of aggregation layers where the main ones are-

- Maximum pooling: selecting the pixel with the maximum value of the elements present in the feature map.
- Average Pooling: Average calculation of the elements present in the feature map.

2.2.3 Fully connected layer

Connects every neuron in one layer to every neuron in another layer. In addition, the image is usually classified by using the SoftMax function according to input from the previous layer.

SoftMax - get as input a value between 0-1 and normalizes it into new vectors that give a total sum of 1.

2.3 Activation function

The activation function calculates the weights of the neuron's input and adds an additional bias to it, thus deciding whether to activate the neuron or not. The purpose of the activation function is to make the neuron's output non-linear.

2.3.1 ReLU Function

The rectified linear unit (ReLU) is the most common activation function used in deep learning models. When the input value is negative the function will return 0, otherwise it will return the positive value back.
The function can be defined as follows: f(x)=max(0,x).

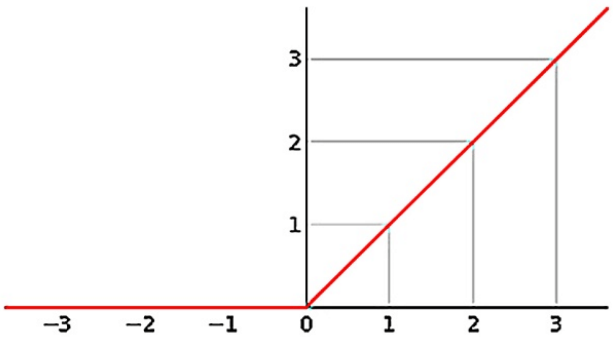


Fig. 1. ReLu activation function

2.4 Flatten layer

This is a layer in a neural network that changes the output from the convolutional layers to a one-dimensional vector which is used as an input to the fully connected layer.

2.5 Softmax

The softmax function transforms the outputs of the neural network into a vector of probability distributions over the input classes. That is, given N classification classes, the softmax operation will return a vector of length N entries, where each entry i in the vector represents the probability that a certain input belongs to class i.

2.6 Recurrent Neural Network (RNN)

It is a type of neural network designed to process continuous data.

An important feature of the RNN is its ability to remember information from previous time steps and use that information to inform its current output.

RNN networks consist of a series of recurrent layers that each have a set of weights and biases that are common to all time steps.

At each time step the input is processed by the recurrent layer and the output is passed to the next time step in addition to the internal state in the network.

2.7 Long short-term memory (LSTM)

Long short-term memory (LSTM) is a type of RNN designed to overcome the limitations of RNNs in processing continuous data when it comes to long-term dependencies.

The network consists of cells similar to RNN but with a more complex structure, where each cell contains a group of gates that control the flow of information into and out of the cell and allow the network to selectively remember or forget information from previous time steps. This allows the network to retain information over a longer period of time.

2.8 Visual Geometry Group 16 (VGG16)

It is a CNN architecture designed for image classification tasks.

The number 16 represents the total number of layers of the network layers, which contains 13 convolutional layers and 3 fully connected layers. In this

architecture there are 3x3 convolutional filters. The use of small filters enables deeper network architectures.

This architecture contains the following layers:

- Input layer - RGB image.
- 13 convolutional layers - each layer is followed by a ReLU activation function and a 2x2 maximum pooling operation. These layers have filter sizes and a different number of filters in each layer that gradually increase the depth of the network. The convolutional filters of VGG use the smallest possible receptive field of 3×3
- ReLu Operation - ReLU is a linear function that provides a matched output for positive inputs and a zero output for negative inputs. VGG has a set convolution step of 1 pixel to preserve spatial resolution after convolution (the step value reflects how many pixels the filter "moves" to cover the entire image space).
- Hidden layers - All hidden layers of VGG networks use ReLU. The latter increases training time and memory consumption with little improvement to overall accuracy.
- Pooling Layers – A pooling layer follows several convolutional layers – this helps to reduce the dimensionality and number of parameters of the feature maps generated by each convolution step.
- Fully connected layers - 3 fully connected layers, where the final fully connected layer contains the number of units that represent the number of classes in the data set on which the network was trained.
- Softmax layer - The output layer uses a Softmax operating function to produce the probability distribution over the classes.

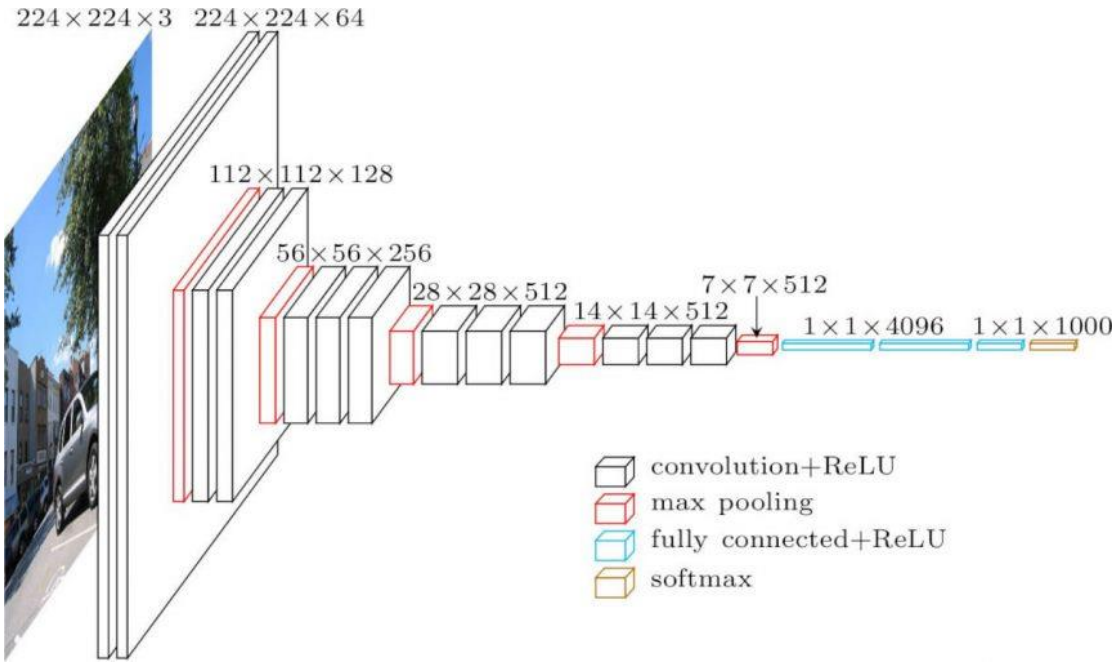


Fig. 2. The architecture of VGG16.

2.9 Decision Tree

A decision tree is a supervised learning algorithm, which is used for classification tasks. The tree consists of a root node, branches, internal nodes and leaf nodes.
A decision tree starts from the root node, the branches coming out of the root node connect to the internal nodes, which are called decision nodes. According to the available features, the nodes form subgroups which eventually reach the final nodes which are the leaf nodes. The leaf nodes represent all possible outcomes.

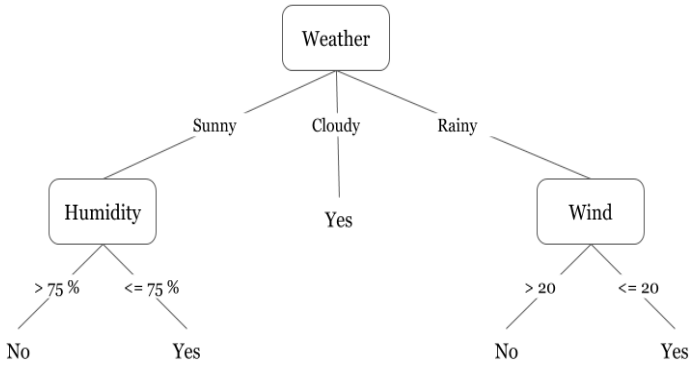


Fig. 3. An example of decision tree

3. Expected Achievements

Our expectation from the project is that we will be able to diagnose a fruit, identify the type of fruit and finally show the user the level of ripeness of the fruit. To classify the type of fruit and to determine if the fruit is considered ripe or not, we will use the VGG16 architecture as in [26].
Then, in order to measure the ripeness of the fruit and watch the passing days until it rots, we purchased green bananas from the supermarket. We took about 100 photos every day, documenting the metamorphosis of the bananas until the bananas rotted. Again, we used the VGG16 model, but this time we used it to estimate how much time a banana has left until it rots.

4. Research

4.1 Dataset

In this section we will describe the datasets we used in this project.
First, to classify the type of fruit and to determine whether it is ripe or rotten we used the following two datasets:

1. https://www.kaggle.com/sriramr/fruits-fresh-and-rotten-for-classification
2. https://www.kaggle.com/moltean/fruits

These datasets contain approximately 26,054 images of different types of fruit, both ripe and rotten.
These datasets contain fruits such as apples, oranges, bananas, avocados, kiwis, mangoes and more. We allocated 80% of the images for training, reserving 10% for validation and another 10% for testing.
Then, in order to predict the level of ripeness of the fruit and since there is no suitable dataset for this purpose, we had to choose a fruit that we could follow for several days. In figure 5 it appears that the time a banana can be stored until it rots at room temperature is between 5 and 7 days. This is a reasonable amount of time to follow a banana in order to create a data set suitable for our purpose. Therefore, we purchased a bunch of green bananas from the supermarket and for 11 days we took about 100 photos every day at different times of the day and created a dataset containing the ripeness of the banana during each day of the 11 days that were recorded.
Again we used 80% of the images for the train, 10% for validation and 10% for tests.

		LOCATION	NOTES	STORAGE DURATION
FRUIT	APPLES ***	Refrigerator	Absorbs odors. Avoid storing with onions, garlic	1 to 3 Months
	BANANAS **	Room Temp.	Avoid storing with high ethylene producers	5 to 7 Days
	BERRIES *	Refrigerator		7 to 10 Days
	CITRUS *	Room Temp.	Avoid storing with high ethylene producers	2 to 3 Weeks
	GRAPES *	Refrigerator	Avoid storing with high ethylene producers	1 to 2 Weeks
	MELONS *	Cool	Avoid storing with high ethylene producers	2 to 3 Weeks
	PEARS ***	Refrigerator	Avoid storing with high ethylene producers	2 to 3 Days (when ripe)
	STONE FRUIT ***	Refrigerator	Avoid storing with high ethylene producers	1 to 2 Weeks

Fig. 4. storage duration of fruits.

4.2 Process

At the beginning of the research for the implementation of our project, we realized that we have to divide the model into two phases, where in the first phase we have to build and train a model that will classify images of fruits according to the type of fruit and in the second phase we have to build and train a model that will classify and return information about the ripeness of the fruit. Thus, we conceived the algorithm described below.

4.2.1 First stage - Classification fruits images according to the type of fruit:

According to [1], the authors of the article suggest performing a process of image enhancement before using a model that will classify the fruit images. They suggest enhancing the image because a low-quality image can hide important details and image enhancement can help classify the fruit effectively.

There are several techniques for enhance an image, we will mention the techniques we studied:

1. Adaptive Trilateral Contrast Enhancement (ATCE)
2. Brightness Preserving Dynamic Histogram Equalization (BPDHE)
3. Gamma Correction (GC)
4. Contrast Limited Adaptive Histogram Equalization (CLAHE)
5. Fuzzy Type-II

After reading [1] and [22], and after researching the image enhancement techniques, it seems that using Fuzzy Type-II offers improved image quality compared to the other image enhancement techniques (ATCE, BPDHE, GC and CLAHE).

The problem with the ATCE, BPDHE, GC and CLAHE techniques is that these techniques fail to address enhancement problems during histogram calculation, these techniques use a global contrast enhancement that does not improve all areas and the complexity of the calculations is high and this may make it difficult to perform the algorithm.

We will use the following image enhancement algorithm using Fuzzy Type-II:

1. Input a fruit image.
2. Initialize L = 256, where L represents 256 gray level values.
3. Calculate histogram, normalize histogram and calculate the probability of the histogram normalization by using the cumulative distribution function (CDF).
4. Calculating a first-order fuzzy moment (a measure of the center of mass of a set of values with a membership degree between 0 and 1, where membership degree is used as a weight), this value is used to describe a fuzzy set-in term of a single value.

   Calculate the first-order fuzzy moment (m) by using the following formula:

$$m = \sum_{i=0}^{L=1} t(i) * \widehat{h_I}(i)$$

$\widehat{h_I}$ - The probability of the histogram normalization

t(i) - Calculate by using the following formula:

$$t(i) = \frac{i}{L-1}$$

i - Gray level i.

5. Calculate membership function by using the following formula:

$$\mu_I(ij) = \frac{L_{ij} - L_{min}}{L_{min} L_{max}}$$

Lmin - Minimum gray level value.

Lmax - Maximum gray level value.

Lij - Gray level of pixel ij, i = 0, 1, 2, …, N and j = 0, 1,2, …, M

6. Calculate histogram.
7. Calculating the probability of a gray level occurring in an image, by counting the number of pixels that have a gray level in the image and then dividing by the total number of pixels in the image.
8. Normalize histogram.
9. Initialize e = 1 - m, ϵ is fuzzy safe entropy, meaning it is a measure of the uncertainty or randomness of a fuzzy set.
10. Obtaining the optimal α and ϒ values by performing the following actions:

for$\alpha = 1$ to $\frac{254}{255}$, compute $\lambda = \frac{1-2\alpha}{\alpha^2}$

Using the following membership function:

$$\mu = \begin{cases} \mu_{dark} & \frac{1}{\alpha}x^2 \quad if\ x\,\varepsilon\,[0,\alpha] \\ & 0 \quad if\ x\,\varepsilon\,(\alpha,1] \\ \mu_{bright} & 0 \quad if\ x\,\varepsilon\,[0,\alpha] \\ & \frac{1-\frac{1}{\alpha}(\frac{1-x}{1+\lambda x})^2}{1+\frac{\lambda}{\alpha}(\frac{1-x}{1+\lambda x})^2} \quad if\ x\,\varepsilon\,(\alpha,1] \end{cases}$$

We want to segment the image into dark and light pixels, where the meaning of 0 is a dark pixel and the meaning of 1 is a bright pixel, and therefore G describes two fuzzy sets, one dark set and one bright set.

$$G \in \left[0, \frac{1}{255}\frac{2}{255}\frac{3}{255}\frac{255}{255}\right]$$

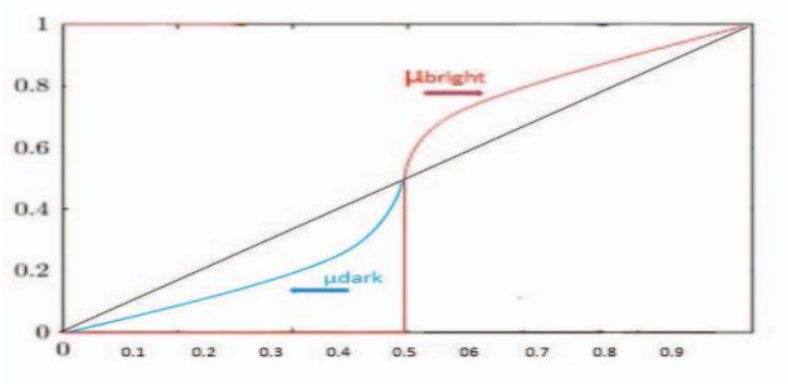


Fig. 5. "The membership function μ_{dark} and μ_{bright} " [From [22] page 4]

Calculation of probability functions for dark and light Fuzzy events, mark these values by M(d) and M(b).
If the current M(d) is greater than the maximum M(d), replace it with the current M(d), same as above for M(b).

11. Modifying the value ϒ according to the following formula:

$$\epsilon = \begin{cases} \frac{p(bright)max}{2} & if m \leq 0.5 \\ \frac{p(dark)max}{2} & otherwise \end{cases}$$

The histogram may be higher if m is less than 0.5 or lower if m is greater than 0.5.

12. To obtain optimal α and ϒ values return to step 10.

According to [1] and [22], it can be seen in the following images that the fuzzy type-II algorithm achieves a better improvement compared to the other techniques.

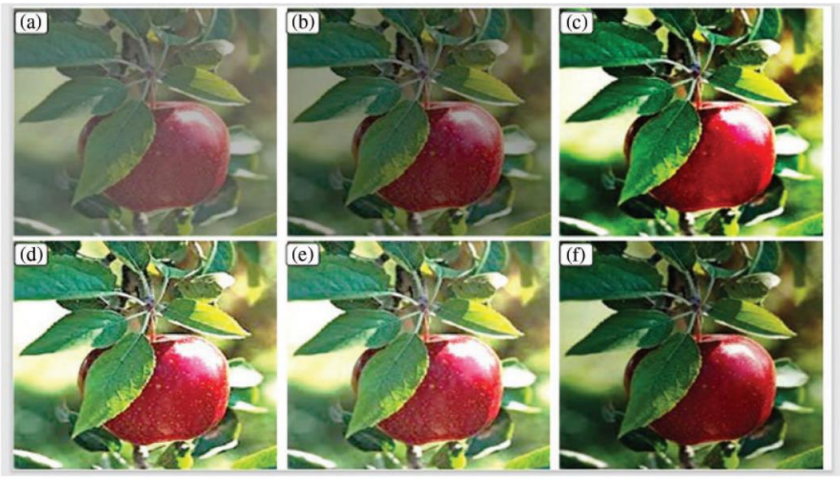


Fig. 6. "Fruit image visual analysis [28–30] (a) input image, (b) (b) enhanced by ATCE, (C) enhanced by BPDHE, (d) enhanced by GC, and (e) by CLAHE, and (f) by Type-II Fuzzy" [From [1] page 9]

Fig. 7. “Results of green apple image, (a) Input (b), (c), (d) (e) and (f) are ATCE, BPDHE, GC, CLAHE and Proposed Scheme output” [from [22] page 6]

4.2.2 Fruit Image Classification model:

There are several models that are used to classify images such as Support Vector Machine (SVM), Feed-forward Neural Network (FFNN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Adaptive Neuro-Fuzzy Inference System (ANFIS).

After reading [1] and [23], we have seen that a combination of the CNN, RNN and LSTM models gives better performance than the models we mentioned above.

CNN extracts specific features from the fruit images, RNN improves the classification of the labels according to coarse and fine categories, LSTM defines the optimal parameters during fruit classification.

After input the enhanced fruit image to the CNN network, convolution layers integrating a RELU activation function were used to extract the features, pooling layers used to reduce the computation time, a flatten layer to convert the extracted features into a one-dimensional vector and a fully connected layer to combine the extracted features.

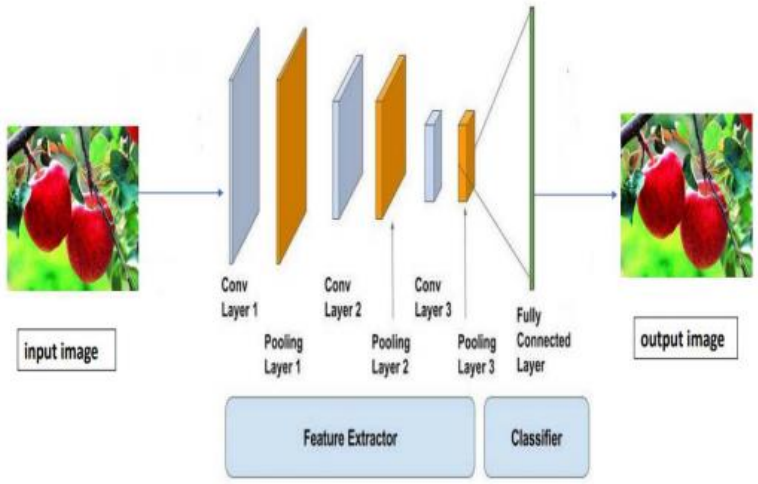


Fig. 8. “Fruit image feature extraction using CNN” [From [23] page 6]

The combination of CNN -RNN -LSTM means that the CNN is used to obtain optimal features, RNN is used to obtain labels and speed up the classification rate and LSTM incorporates a memory cell in order to deal with the vanishing gradient problem.
The RNN is integrated in the model with the CNN to overcome the limitations of the CNN in fruit classification.

Features extracted using CNN and labeled using RNN are fed into LSTM for fruit classification.

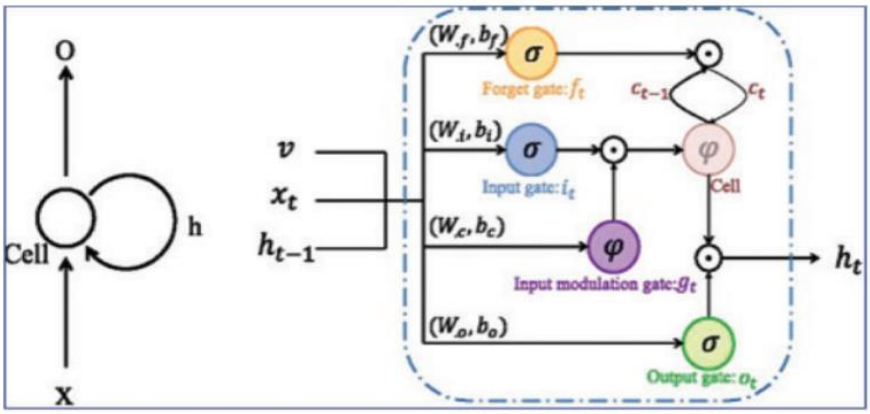


Fig. 9. “RNN (left) and LSTM (right) collaboration” [From [1] page 12]

Details of the stages of the model:
Step 1 - Features will be extracted using the convolution layer, linear layers and fully connected layers of CNN. The extracted properties are based on strength, texture and shape.

Step 2- The features will be labeled by "coarse" and "fine" categories using RNN.
Step 3 - A filtering process used by the LSTM is done and with the help of soft-max-layer the fruits are classified.

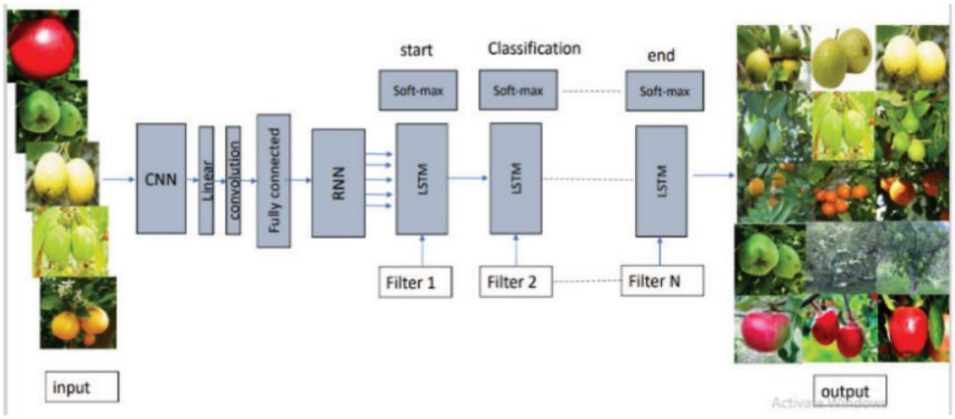


Fig. 10. “Fruit classification model” [From [1] page 13]

4.2.3 Second stage - Classification fruits images according to ripeness level:

After reading the article [2] we realized that there are several options to classify the fruit according to its ripeness and we checked what would be the best method that would achieve the best classification results. The possible methods:

1. Naive Bayes classifier:

Naive Bayes classification is a collection of classification methods based on Bayes' law and the assumption that there is no dependence between the properties of the classified objects when their classification is already known.

Using Bayes' Law, we find the probability that A will happen, given that B has happened. $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$ Here, B is the evidence and A is the assumption. The assumption is that the attributes are independent - that is, one attribute does not affect the other. That's why it's called naive.

2. Artificial Neural Network (ANN):

Artificial neural network is a mathematical model that includes calculations that simulate brain processes similar to the processes that occur in a natural neural network in the human brain. (There is an additional explanation above in the background for Neural Network).

3. Decision Tree:

Decision tree is from the field of statistics, like a tree used as a prediction model. A decision tree is a binary tree that maps observations to an item and

creates inferences about the target value of the item. The tree consists of decision nodes where a condition is tested on a certain characteristic of the observations and leaves that contain the predicted value for the observation corresponding to the route that reaches them in the tree. (There is an additional explanation above in the background).

In a decision tree, to classify fruits as ripe or unripe, we need to extract different features from the images such as color, shape, texture in order to have nodes in the tree. The decision tree may contain many features and extracting these features from images will require some image processing techniques such as color analysis, object detection, and more.

A simple tree for example:

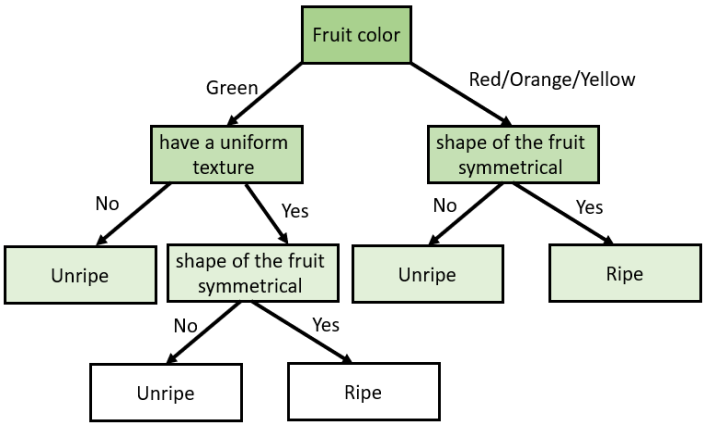


Fig. 11. An example of a simple tree

In fruit identification, color is an important feature because it allows us to evaluate the level of ripeness of the fruit and also indicates defects that may be in the fruit.

We will measure the color level using the basic color components - red, green and blue by using equations –

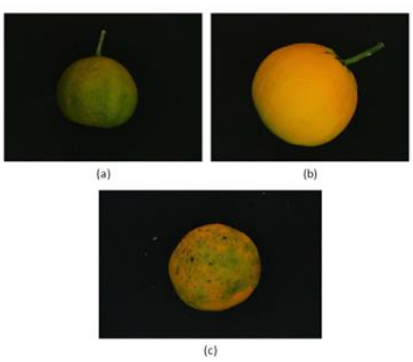
$$\mu_r = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} f_r(i,j)$$

$$\mu_g = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} f_g(i,j)$$

$$\mu_b = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} f_b(i,j)$$

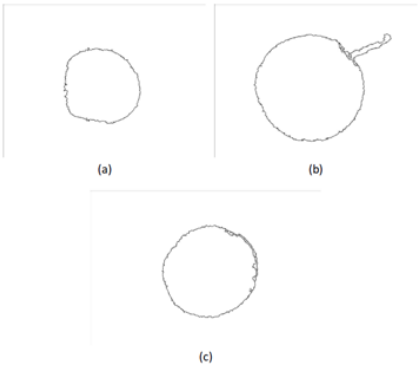
(From [2] page 3)

In addition, to extract the feature of the fruit borders, a border/interior pixel classification (BIC) is used.



Original images

Fig. 12. (From [2] page 3)



After BIC

Fig. 13. (From [2] page 3)

After extracting the features, the tree must be trained and the decisions it makes must be evaluated by measures such as precision, recall, and more.

If the result that comes out of the tree will not be as good as we expected, additional features must be added or some of the existing ones must be replaced.

After in-depth research and after reading the articles [24] and [25] we discovered that it is possible to classify the type of fruit and also classify whether it is ripe or rotten using a deep learning model. Following this, we delved into the deep complexities of the models presented below.

1. **VGG16**
   A convolutional neural network architecture consisting of 13 convolutional layers and 3 fully connected layers, with 3x3 filters throughout the network. VGG16 is often used as a base model and has achieved good performance in various image classification tasks.

2. **ResNet50**
   ResNet50: ResNet50 is a residual neural network architecture that introduced the concept of residual learning. It has 50 layers, consisting mainly of convolutional layers with shortcut connections that allow direct flow of information from earlier layers to later layers. This helps reduce the vanishing gradient problem and allows training of deeper networks.

3. **MobileNet**
   A lightweight convolutional neural network designed for mobile and embedded devices with limited computing resources. It uses depth-separable convolutions, which split the standard convolution into depth and point evolution, reducing the number of parameters and computational cost. MobileNet achieves a good trade-off between accuracy and model size, making it suitable for resource-constrained environments.

4. **InceptionV3**
   An architecture that uses a combination of different filter sizes (1x1, 3x3 and 5x5) within the same layer to capture features at different scales. This allows the network to efficiently learn both local and global features. InceptionV3 is widely used for image classification and object detection tasks.

5. **DenseNet121**
   It is a densely connected convolutional neural network that addresses the issue of information flow and gradient elimination by introducing dense connections between layers. In DenseNet, each layer receives direct input from previous layers, enabling feature reuse and encouraging feature propagation. DenseNet121 has 121 layers.

6. **InceptionResNetV2**
   Combines the Inception module and residual connections from ResNet. It leverages the advantages of both architectures to create a deep neural network with improved accuracy. InceptionResNetV2 achieves high performance in image classification tasks while maintaining a number of manageable parameters.

7. **Xception**
   It is an extension of the Inception architecture that replaces the standard inception modules with depth-separable convolutions. By making extensive use of depth-separable convolutions, Xception aims to effectively capture both spatial and channel dependence.

8. **CNN**
   Refers to a wide category of neural networks specifically designed to process grid-like data, such as images. It usually consists of convolutional layers, pooling layers and fully connected layers. CNNs exploit the spatial structure of the input data by using joint weights and local receptive fields. They have become the backbone of many image classification models, including those mentioned above.

4.2.4 Architecture performance comparison:
According to an article [24] which examined the performance of the models VGG16, ResNet50, MobileNet, InceptionV3, Desnet121, InceptionResNetV2, Xception and CNN for fruit and vegetable image classification, it can be seen from the table below that VGG16 has better performance and better accuracy compared for the other models.

Model Name	Training Accuracy	Testing Accuracy	Training Time	Precision	Recall	F1
VGG16	99.90%	99.80%	01:36:16	0.998	0.998	0.998
ResNet50	99.80%	89%	01:30:03	0.948	0.914	0.914
MobileNet	99%	69%	00:28:59	0.782	0.704	0.657
InceptionV3	79%	53%	00:53:57	0.627	0.529	0.493
Desnet121	99%	88%	02:04:30	0.93	0.876	0.874
InceptionResNetV2	90.73%	61.05%	01:28:00	0.694	0.607	0.577
Xception	96%	60%	01:38:35	0.772	0.592	0.552
CNN-Conv2D	99%	99%	01:07:30	0.928	0.979	0.979

Fig. 14. Experiment Results Table [24]

According to article [1] and fig.16 and fig.17 it can be seen that classification of fruit images according to a combined CNN, RNN and LSTM model also shows good performance.

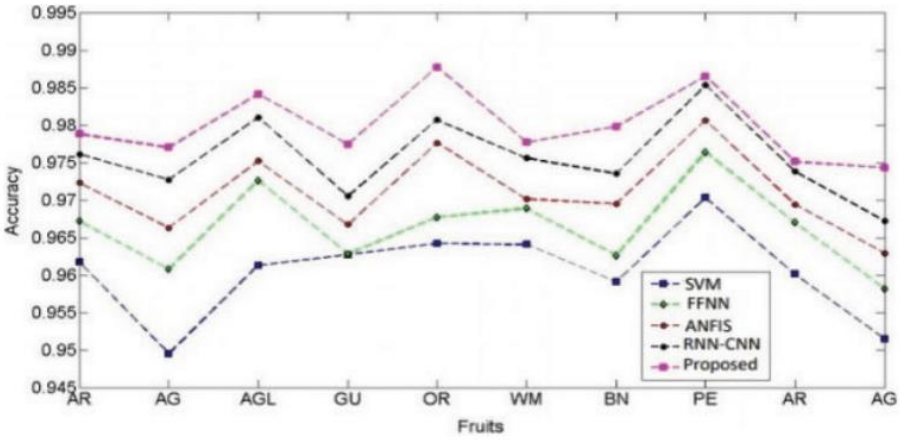


Fig. 15. “Accuracy analysis” [from [1] page 14]

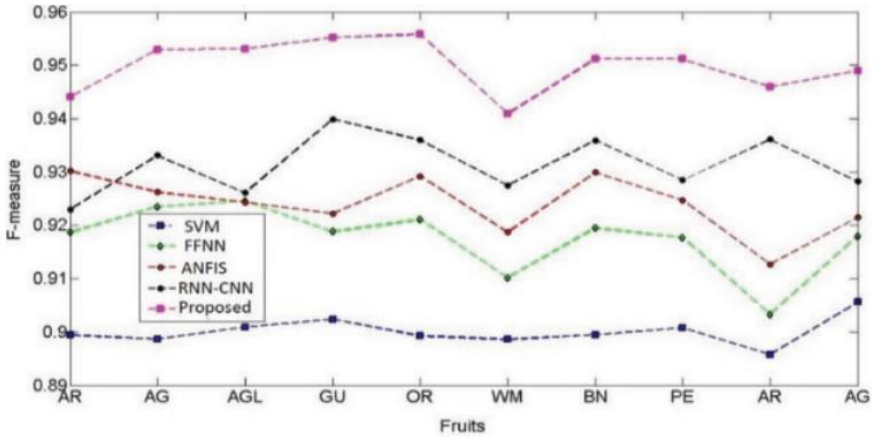


Fig. 16. F-measure [from [1] page 14]

In addition, according to article [25] it can be seen that it is possible to classify both the ripeness of the fruit and to examine whether the fruit is ripe or not by using the

VGG16 model instead of combining CNN, RNN, LSTM and a decision tree, also, we saw in the table above that the accuracy of VGG16 is 99.90% While it can be seen that the accuracy of decision tree in fig.18 is 93.13%.
Since our data set is large and higher accuracy is required we decided to use VGG16. At the same time deep learning models like VGG16 can automatically learn complex patterns and representations from raw image data, and can capture subtle visual cues that indicate fruit ripeness.

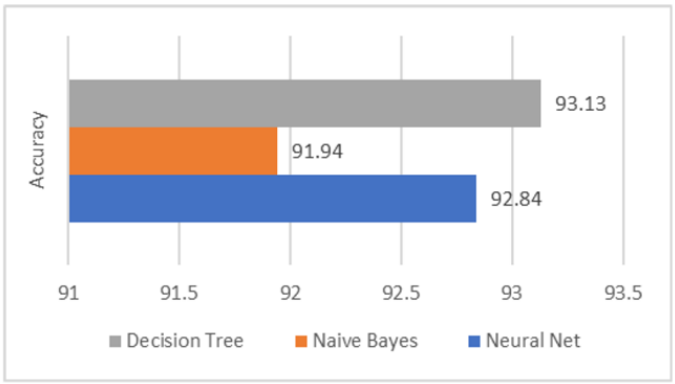


Fig. 17. (From article [2] page 4)

Now that we have decided on the model that is suitable for classifying the images according to the type of fruit and determining whether it is ripe or not, we will present our algorithm.

Step 1
Input a fruit image.

Step 2
Resize the image to 96x96 with a depth of 3.

Step 3
Scale the raw pixel intensities to the range [0, 1].

Step 4
Dividing the images into training and testing splits using 80% of the data for training, 10% for validation and the remaining 10% for testing.

Step 5
Building the image generator for data augmentation that includes the following values: rotation_range=25, width_shift_range=0.1, height_shift_range=0.1, shear_range=0.2, zoom_range=0.2, horizontal_flip=True, fill_mode="nearest".

These values were carefully chosen to create a balance between showing significant variations and avoiding excessive distortions or unrealistic transformations. Smaller values may not provide diversity for the model and larger values may introduce unrealistic transformations that may negatively affect performance.

Step 6

VGG16 network training.

The first layers contain a convolution layer containing 32 filters, where the size of each filter is 3x3, along with a Relu activation function and a max pooling layer, where the pool size is 3x3 and using a dropout of 25%.

The layers after that contain 2 convolution layers each containing 64 filters where the size of each filter is 3x3, along with a Relu activation function and one layer of max pooling, where the pool size is 2x2 and using a dropout of 25%.

Now another 2 convolution layers are used, each containing 128 filters with the size of each filter being 3x3, together with a Relu activation function and one layer of max pooling, with the pool size being 2x2 and using a dropout of 25%.

Then, to use softmax, a fully connected layer is used together with a Relu activation function and a dropout of 50% is used.

And then to return the predicted probabilities for each class label the softmax layer is used.

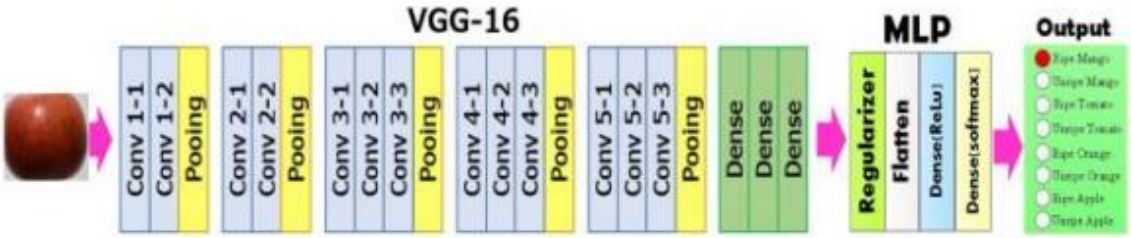


Fig.18. Architecture using VGG16 (From article [25] page 4)

We used this algorithm both to identify the type of fruit and to determine whether it is ripe or not and also to predict the level of ripeness of the banana. We use the VGG16 model to classify bananas based on their visibility on a given day, using a dataset created specifically for bananas and used to train the model.

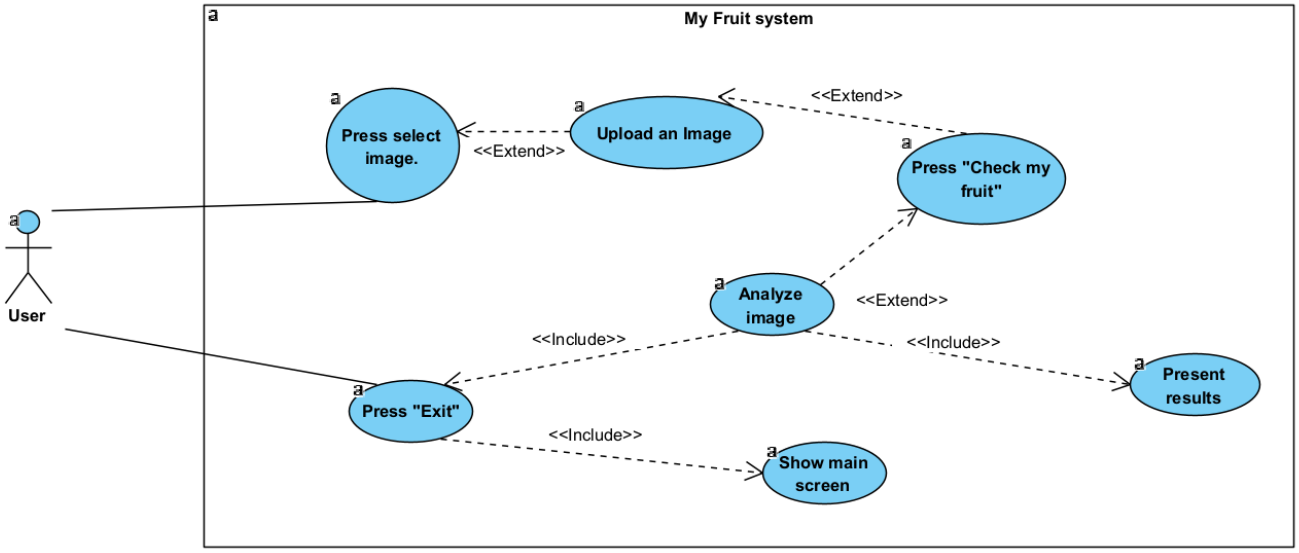
So, if a banana is unripe, using this classification, we can predict how many days the banana will be ripe, and if the banana is already ripe, it will be possible to predict how many more days the banana will be edible.

4.3 Product

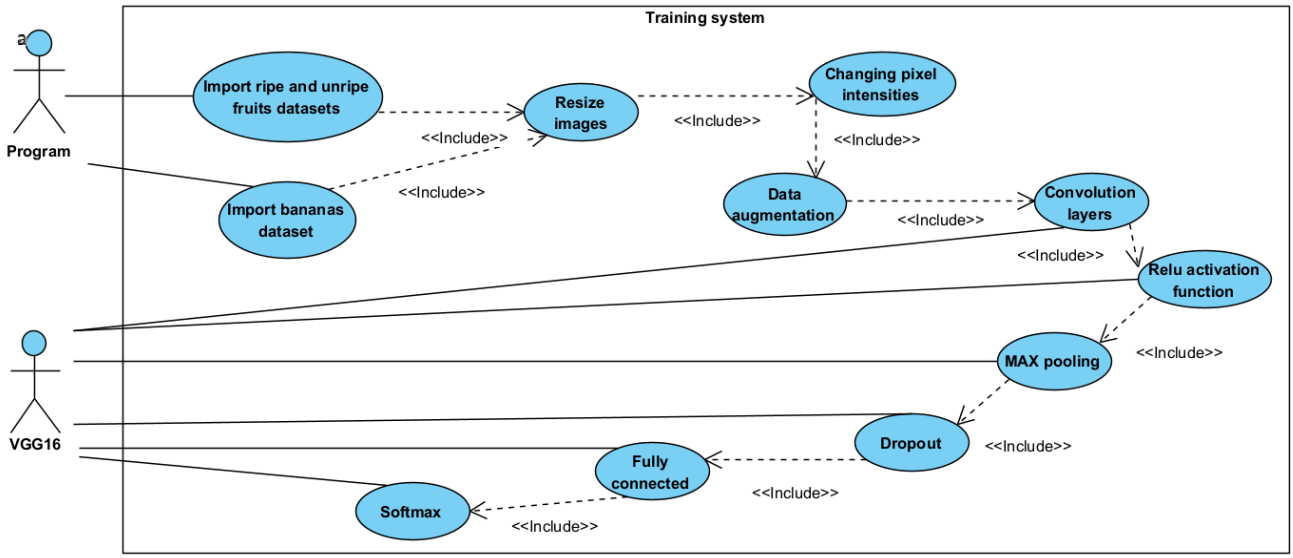
4.3.1 UMLs

4.3.1.1 Use Case:

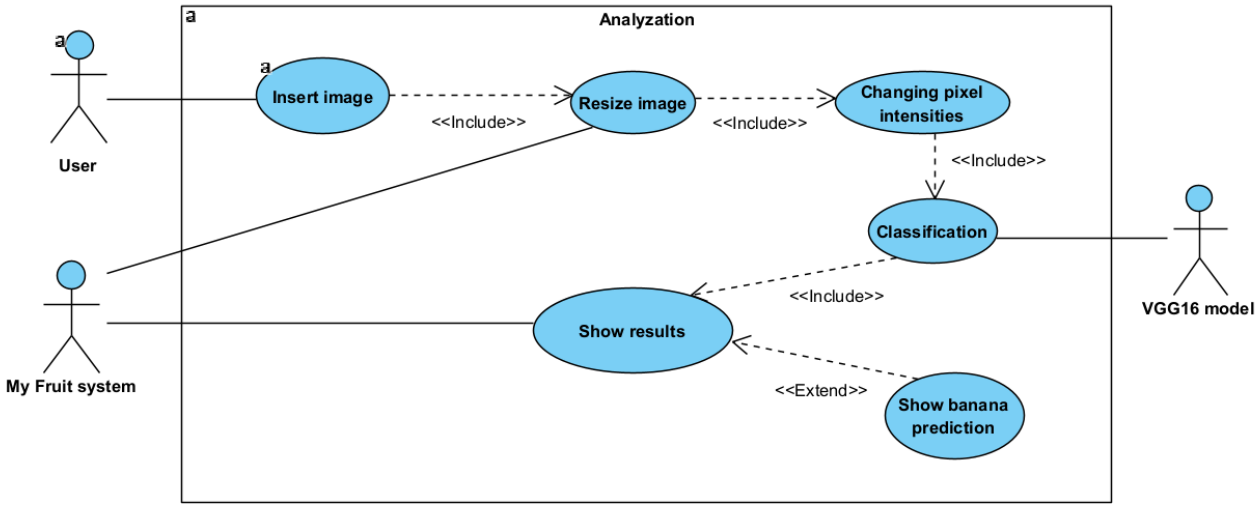
Ripeness Fruits System:



Training:

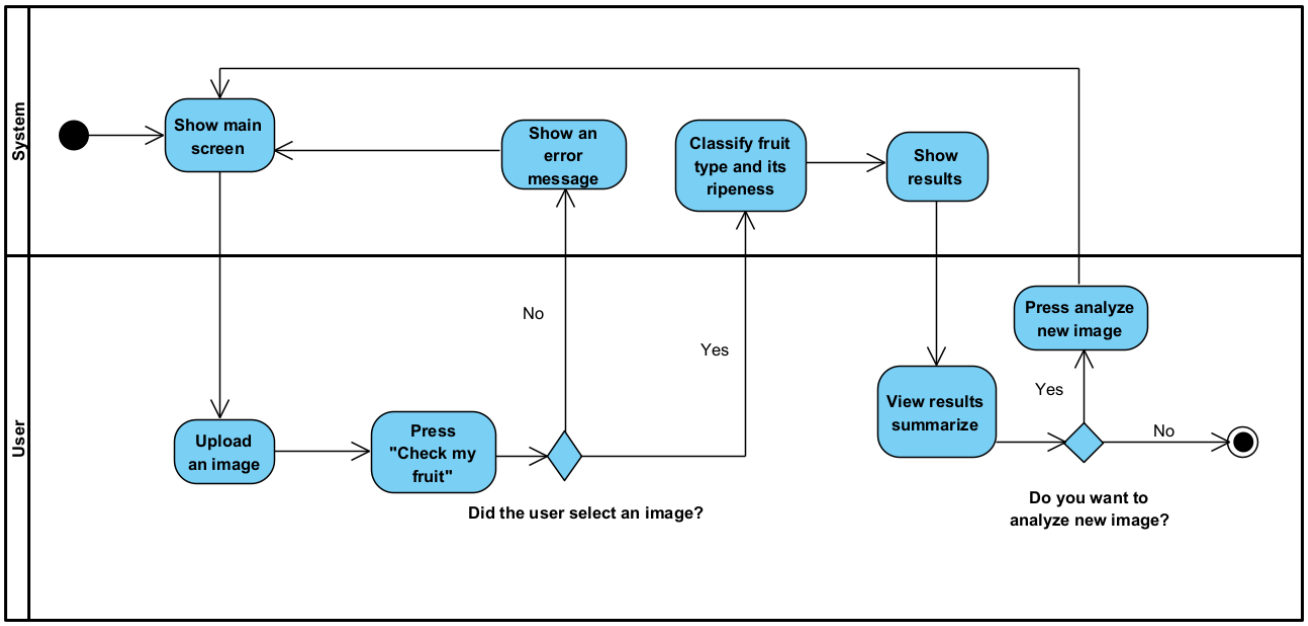


Analyzation:

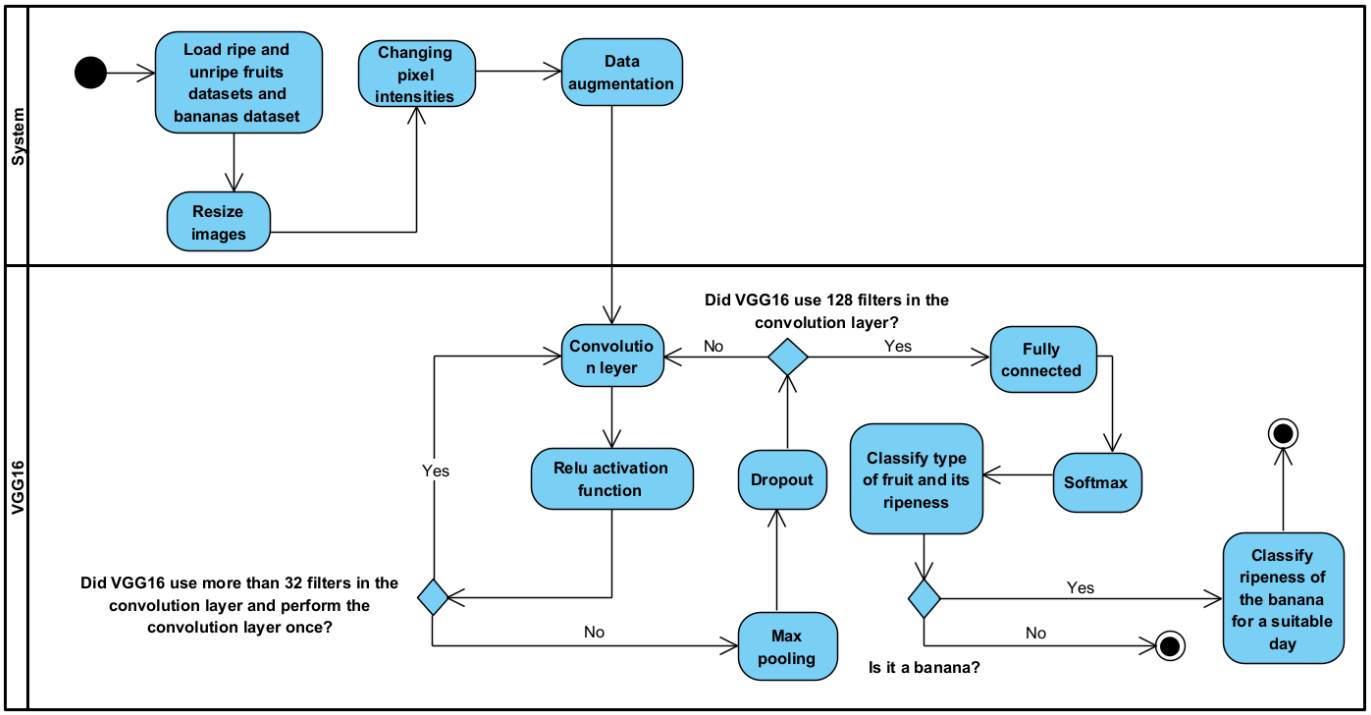


4.3.1.2 Activity:

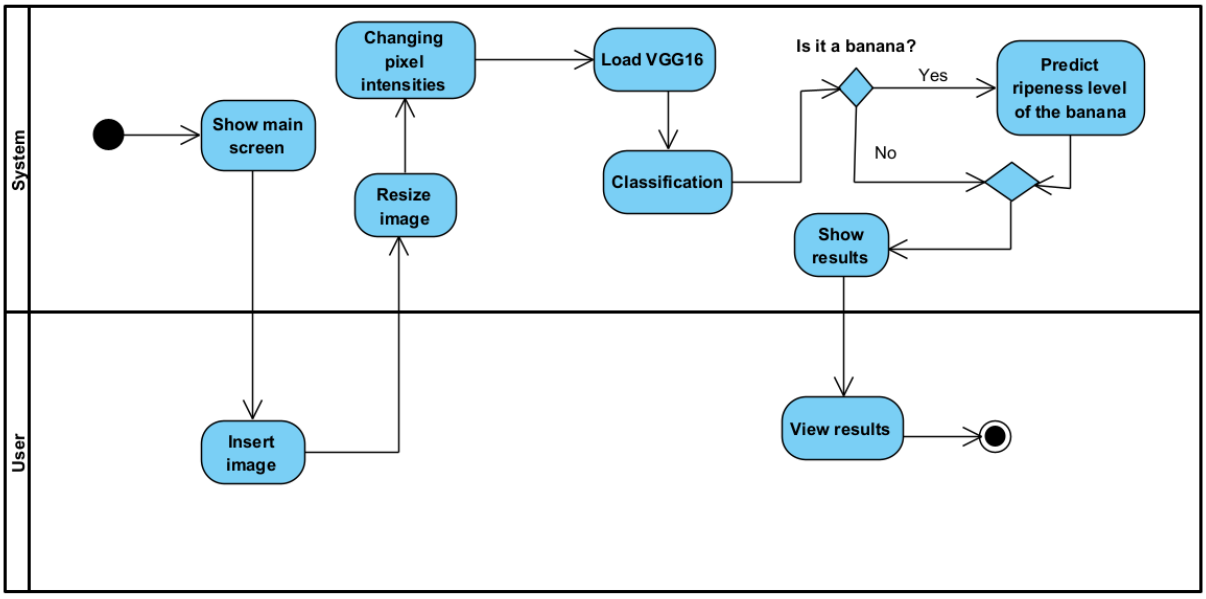
Ripeness Fruits System:



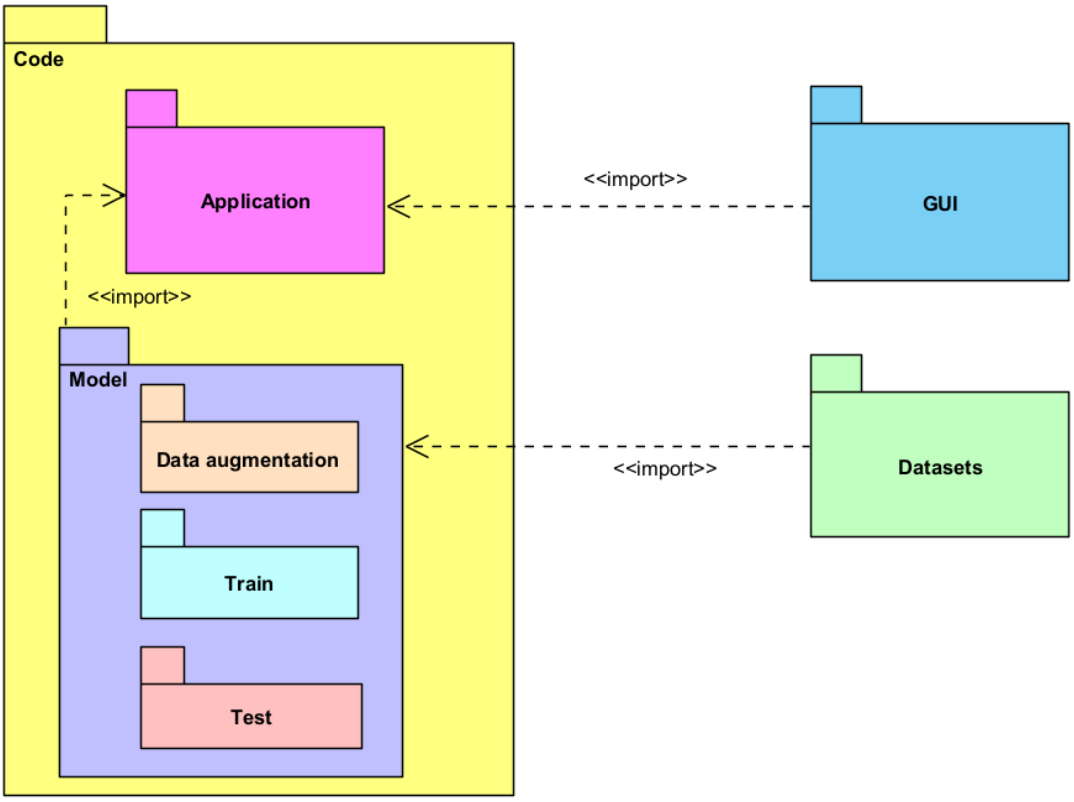
Training:



Analyzing:

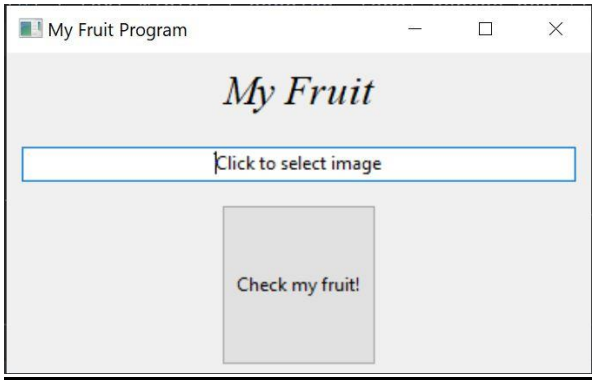


4.3.1.3 Package:



4.3.2 Screens

4.3.2.1 Main Screen:

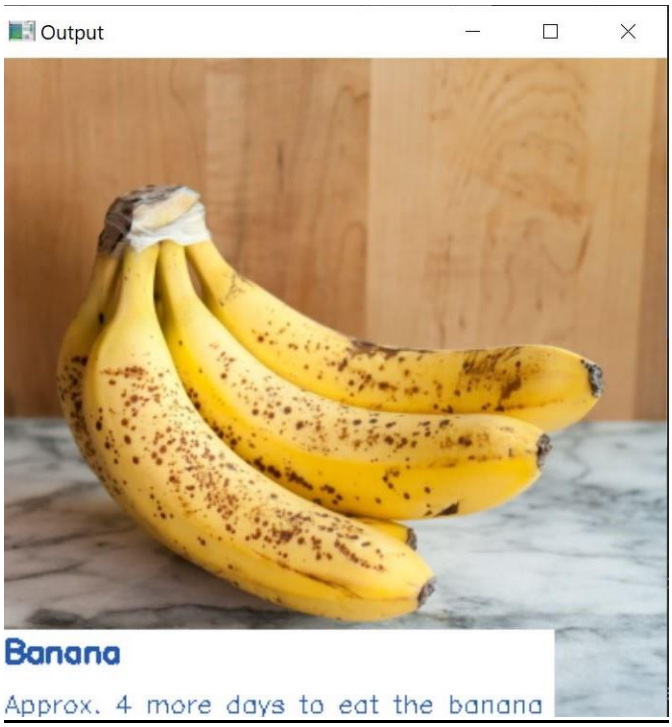


4.3.2.2 Result for a ripe apple:

4.3.2.3 Result for a rotten apple:

4.3.2.4 Result for unripe banana:

4.3.2.5 Result for a ripe banana:



4.3.2.6 Result for a rotten banana:

5. Results

We loaded a library called keras that we used in building the neural network of VGG16.

We trained our model with 100 epochs, batch size of 32, learning rate of 0.001, with each image size 96X96X3.

Accuracy:

This is a way to evaluate the performance of the model. The accuracy of the model describes the number of classifications the model was able to correctly predict divided by the total number of predictions made.

This can be seen by calculating the following formula.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

True positive (TP) - cases that are expected to be positive and are actually positive.

True negative (TN) - cases that are expected to be false and are actually false (do not belong to a certain class).

False Positive (FP) - cases that are expected to be positive (belong to a certain class) but are actually negative (do not belong to the class).

False negative (FN) - cases that are expected to be negative (do not belong to a certain class) but are actually positive (belong to a class).

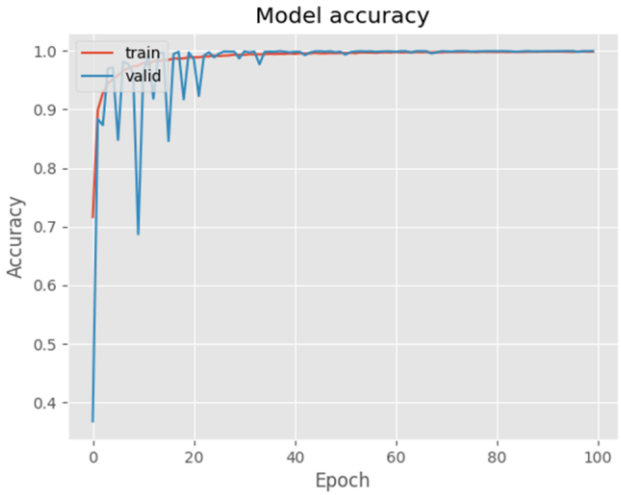


Fig.19. Train and validation accuracy results for fruits ripe or rotten model

Our model reached training accuracy of 99.51% and 98.91% for validation.

Training accuracy is the percentage of correct predictions the model makes on the data it was trained on. In our case, it's 99.51%, meaning the model predicts the training data extremely well.
Validation accuracy is the percentage of correct predictions the model makes on new, unseen data (the validation set). In our case, it's 98.91%, meaning the model also performs very well on data it wasn't trained on.
In summary, these percentages indicate our model is performing excellently on both the training and validation data, suggesting a well-trained model.

Regarding the training of the model for predicting the ripeness stage of the bananas, the accuracy can be seen in the diagram below.

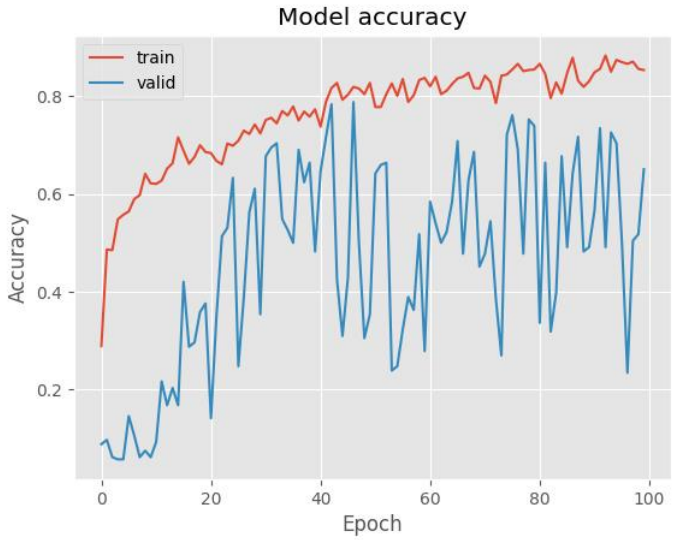


Fig.20. Train and validation accuracy results for predicting the ripeness stage of the bananas

Our model reached training accuracy of 85.32% and 65.04% for validation.

Precision:

Precision is a measure of how many of the model's positive predictions were actually correct. This can be seen by calculating the following formula.

$$Precision = \frac{TP}{TP + FP}$$

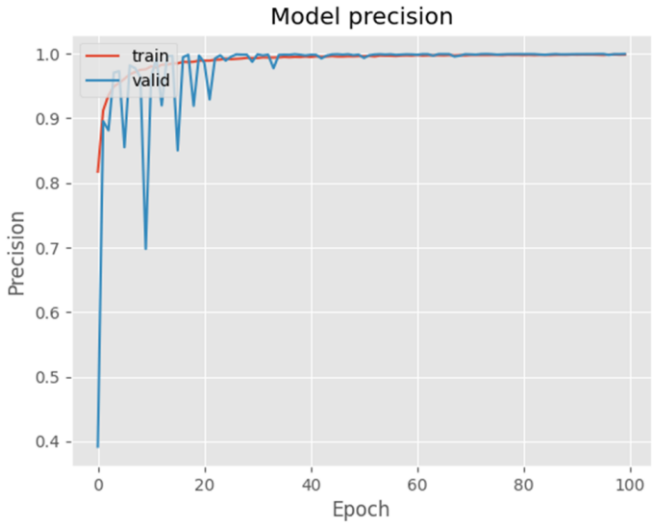


Fig.21. Train and validation precision results for fruits ripe or rotten model

Our model reached training Precision of 99.52% and 98.91% for validation.

Our model's training precision of 99.52% means that when it predicts a positive outcome on the training data, it's correct about 99.52% of the time.
The validation precision of 98.91% means that when it predicts a positive outcome on new, unseen data, it's correct about 98.91% of the time.
These high percentages suggest our model is very good at making accurate positive predictions.

Regarding the training of the model for predicting the ripeness stage of the bananas, the precision can be seen in the diagram below.

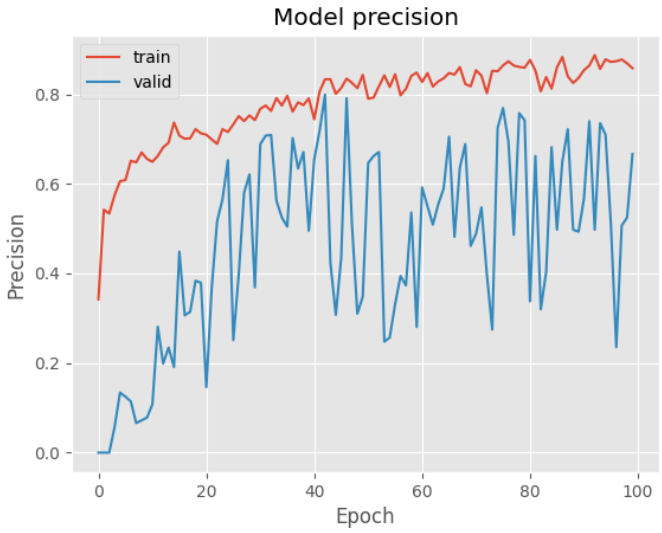


Fig.22. Train and validation precision results for predicting the ripeness stage of the bananas

Our model reached training Precision of 85.83% and 66.67% for validation.

Recall:

Recall is a metric that shows how many of the actual positives the model managed to capture through its predictions.
This can be seen by calculating the following formula.

$$Recall = \frac{TP}{TP + FN}$$

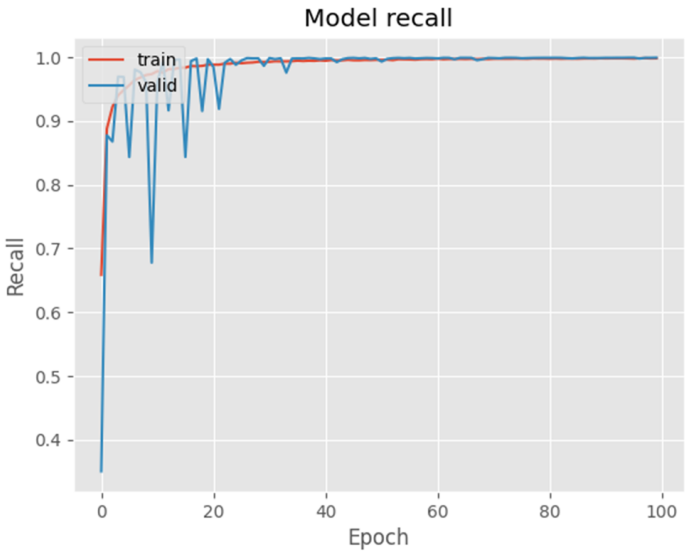


Fig.23. Train and validation recall results for fruits ripe or rotten model

Our model reached a training recall of 99.51% and 98.87% for validation.

A training recall of 99.51% means the model identified 99.51% of the positive cases correctly in the training data.
A validation recall of 98.87% means the model identified 98.87% of the positive cases correctly in the unseen validation data.
In short, our model is highly effective at identifying positive cases in both the training and validation datasets.

Regarding the training of the model for predicting the ripeness stage of the bananas, the recall can be seen in the diagram below.

Fig.24. Train and validation recall results for predicting the ripeness stage of the bananas

Our model reached a training recall of 84.75% and 64.60% for validation.

Loss function:

"Loss" is a number indicating how well the model's predictions match the actual data.

This is a scalar value that we try to minimize during the training of our model. When we manage to reduce the value more, it shows that the predictions are closer to the original labels.

We used the loss Crossentropy Categorical function of keras which calculates the loss between labels and predictions.

This can be seen by calculating the following formula.

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output}\\ \text{size}}} y_i \cdot \log \hat{y}_i$$

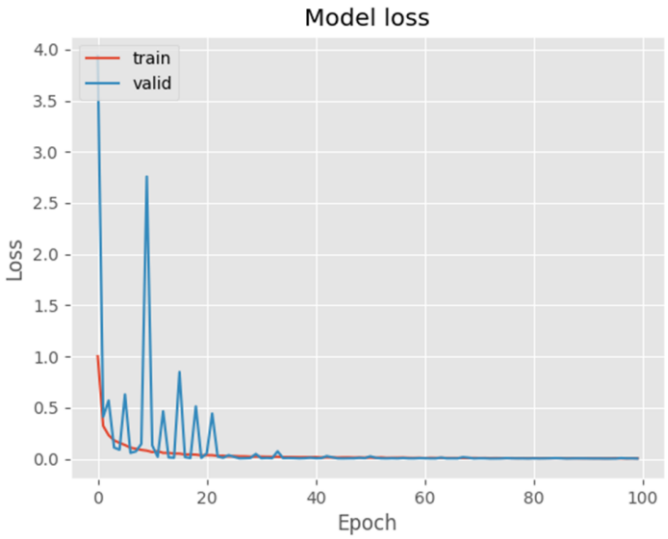


Fig.25. Train and validation loss results for fruits ripe or rotten model

Our model reached a training loss of 0.0174 and 0.0344 for validation.

A training loss of 0.0174 means the model's predictions on the training data were quite close to the actual values.

A validation loss of 0.0344 means the model's predictions on the unseen validation data were also close to the actual values, though not quite as close as on the training data.

our model is performing well, but it's slightly less accurate on new, unseen data compared to the data it was trained on.

Regarding the training of the model for predicting the ripeness stage of the bananas, the loss can be seen in the diagram below.

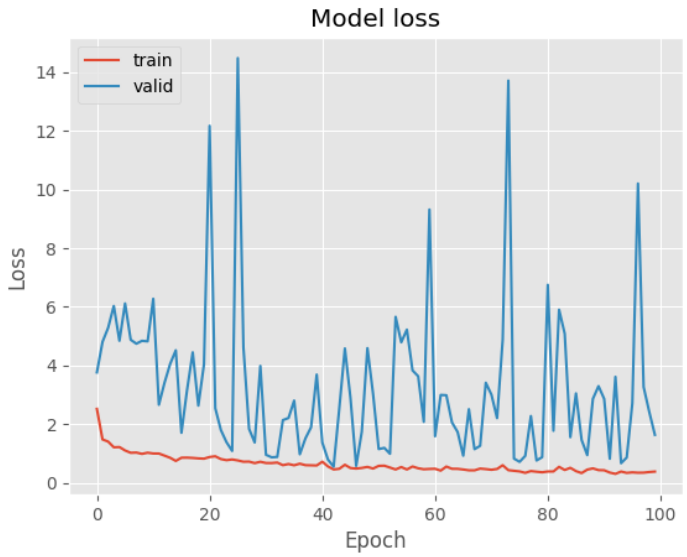


Fig.26. Train and validation loss results for predicting the ripeness stage of the bananas.

Our model reached a training loss of 0.3818 and 1.6352 for validation.

Confusion matrix

A confusion matrix it's a tool for understanding prediction errors, it visualizes and understands the performance of a classification model. It shows the counts of true positive, true negative, false positive, and false negative predictions, helping us to see if the model is confusing two classes.

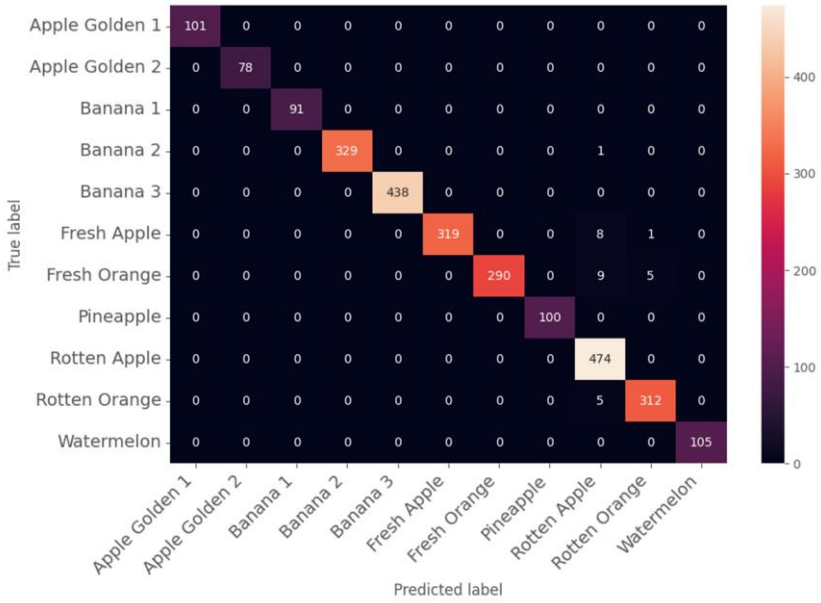


Fig.27. Train and validation confusion matrix results for fruits ripe or rotten model.

From the datasets of the ripe and rotten fruits, it can be seen according to the confusion matrix that 2637 images out of 2666 images were correctly predicted, meaning 98% of the test set were correctly predicted.

Regarding the training of the model for predicting the ripeness stage of the bananas, the confusion matrix can be seen in the diagram below.

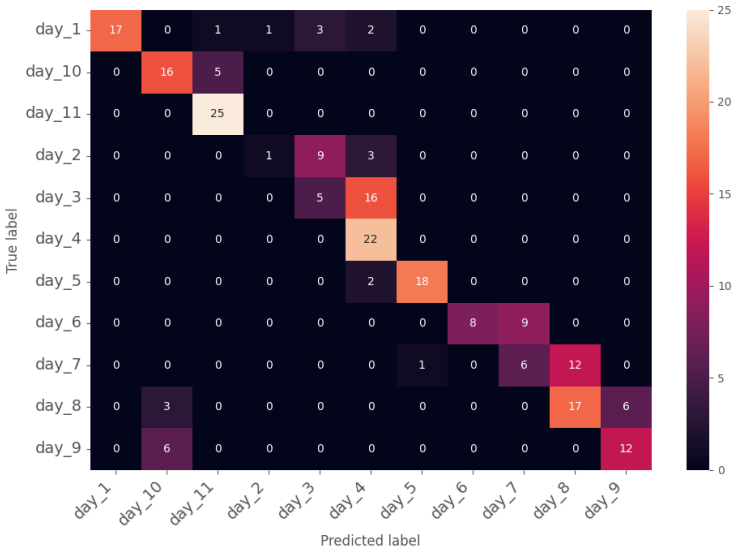


Fig.28. Train and validation confusion matrix results for predicting the ripeness stage of the bananas.

From the dataset of the bananas, it can be seen according to the confusion matrix that 147 images out of 226 images were correctly predicted, meaning 65% of the test set were correctly predicted.

6. User Guide

6.1 Software Environment:

- Windows software
- Python = 3.10.6
- PyCharm

6.2 Running Instructions:

The software system requires the following software requirements:

- python == 3.10.6

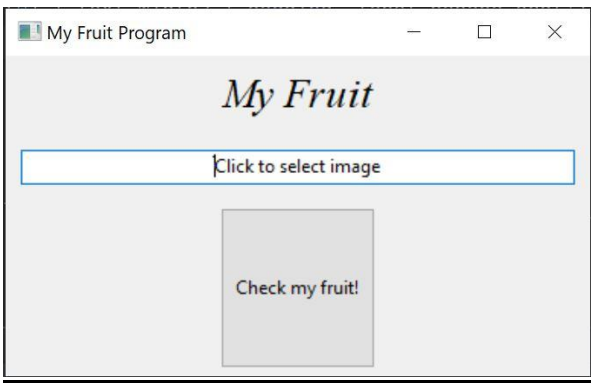
- numpy == 1.23.5
- matplotlib == 3.7.1
- wxPython == 4.2.0
- imutils == 0.5.4
- opencv-python == 4.7.0.72
- keras == 2.12.0
- pandas == 1.5.3
- seaborn == 0.12.2
- sklearn == 0.0.post5

6.3 General Description:

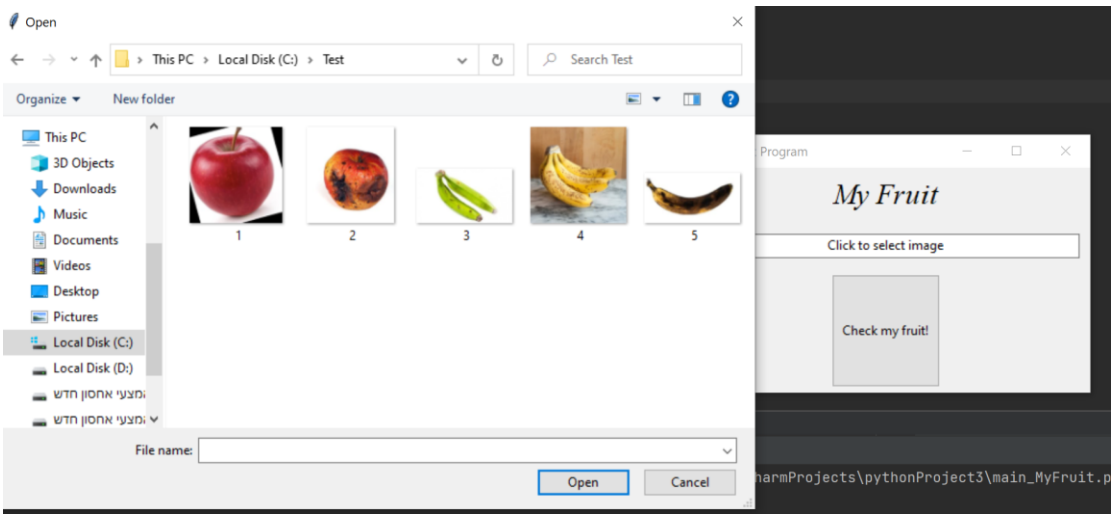
The system is based on deep learning and is designed to help identify in the image which fruit it is and to give an estimate of how much longer the fruit will be ripe, how long it will be ripe for and when it is considered rotten and it is better not to eat it. The system will not identify all the fruits and give an accurate assessment for each fruit, but this tool is a good basis for future development. You can continue training the model with a larger dataset with more fruits in the train_MyFruit.py file and changing the "data_path" function to the path of your dataset and where to save the trained model. Then change the paths in the file test_MyFruit.py. To start the software, run the main_MyFruit.py file.

6.4 User Instructions:

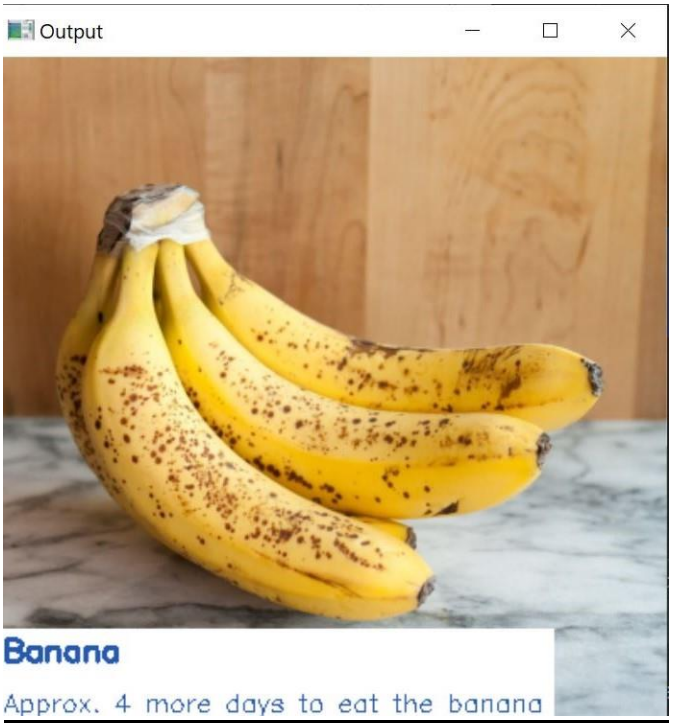
1. run the main_MyFruit.py file:



2. Press the button "Click to select image" and search for fruit image for analyzing.



3. Press the button "Check my fruit" and wait until you see your results.



7. Evaluation Plan

Test ID	Test Description	Expected results
1	User uploads a picture that is not of a fruit	System shows a popup error message. Output: "No fruit detected".
2	User uploads a file that is not an image	System shows a popup error message. Output: "Please only upload a picture of a fruit".
3	User uploads a picture of a ripe apple	System analyzes the image and shows that it is an apple and it is ripe.
4	User uploads a picture of an rotten apple	System analyzes the image and shows that it is an apple and it is rotten.
6	User uploads a picture of a unripe banana	System analyzes the image and shows that it is an unripe banana and how many more days it will be unripe.
7	User uploads a picture of a ripe banana	System analyzes the image and shows that it is a ripe banana and how many more days it will be ripe.
8	User uploads a picture of a rotten banana	System analyzes the image and shows that it is a rotten banana and recommends not eating it.

8. Conclusions

In our research we propose a method for classifying fruits according to their ripeness.

First, we use the VGG16 model to identify the type of fruit and whether the fruit is ripe or rotten.

After that we wanted to check more deeply if it is possible to make a prediction for the user regarding the number of days left for bananas until they are ripe or alternatively the number of days left for bananas until they rot.

For this we created a data set in which we recorded the bananas every day until they rotted.

We then used VGG16 again to predict what day the banana was photographed and based on that estimate how much time it has left to ripen or how long it will take for it to rot.

We are less satisfied with the results we obtained for the prediction of the bananas.

We think that a lot of research should be done in this area in order to achieve better results.

We assume that since there are days when the bananas look the same, it is sometimes difficult for the model to distinguish the exact day the banana was photographed and therefore, it may be worthwhile to create a larger data set and perhaps perform a comparison between different architectures of neural networks where deeper learning is performed.

9. References

[1]Harmandeep singh Gill, Youseef Alotaibi, Osamah Ibrahem Khalaf, Saleh Mohammed Alghamdi, Fruit Image Classification Using Deep Learning, 2022.

[2]Abdul Wajid, Niraj Kumar Singh, Pan Junjun, Muhammad Ali Mughal, Recognition of ripe, unripe and scaled condition of orange citrus based on decision tree classification, 2018.

[3]https://www.geeksforgeeks.org/activation-functions-neural-networks/

[4]https://www.kaggle.com/code/dansbecker/rectified-linear-units-relu-in-deep-learning/notebook

[5]https://www.simplilearn.com/tutorials/deep-learning-tutorial/convolutional-neural-network

[6]https://www.tutorialspoint.com/keras/keras_flatten_layers.htm

[7]https://deepai.org/machine-learning-glossary-and-terms/softmax-layer

[8]https://www.pinecone.io/learn/softmax-activation/

[9]https://www.ibm.com/topics/recurrent-neural-networks

[10]https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[11]https://www.sciencedirect.com/science/article/abs/pii/S0952197620302487

[12]https://www.ibm.com/topics/decision-trees

[13]https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

[14]https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c

[15]https://www.ibm.com/topics/neural-networks

[16]https://www.sciencedirect.com/science/article/pii/S0167865514002852?casa_token=rFVPVZdyYo8AAAAA:JzohJizy-IBJckmv1KHaQID_1smOlMQGGr-R77WIOph_HPWMBZuFAwEwA5XHzwfTm9JBZb8-Zvg

[17]https://ieeexplore.ieee.org/abstract/document/4429280

[18]https://lindevs.com/apply-gamma-correction-to-an-image-using-opencv

[19]https://www.sciencedirect.com/topics/earth-and-planetary-sciences/artificial-neural-network

[20]https://www.sciencedirect.com/science/article/abs/pii/S1568494614004438

[21]https://philarchive.org/archive/ALKCOA

[22]Dr.Harmandeep Singh Gill, Dr.Baljit Singh Khehra, Er.Bhupinder singh Mavi, Fruit images Visibility enhancement using Type-II Fuzzy, 2021.

[23]Dr.Harmandeep Singh Gill, Dr.Baljit Singh Khehra, Fruit Image classification using Deep Learning, 2022.

[24]Hanshu Tomar, Prof.Basel Magableh, Multi-Class Image Classification of Fruits and Vegetables Using Transfer Learning Techniques, 2020.

[25]Jasman Pardede, Implementation of Transfer Learning Using VGG16 on Fruit Ripeness Detection, 2021

[26] https://www.pyimagesearch.com/2018/04/16/keras-and-convolutional-neural-networks-cnns/

10. Git link

https://github.com/shaniFr/fruit-ripeness