

# **FRUITS IMAGE DETECTION USING CNN**

Submitted in partial fulfillment of the requirements for the  
award of Bachelor of Engineering Degree in Computer  
Science and Engineering

By

SENTHIL KUMAR.B ( Reg. No. 37110694)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
SCHOOL OF COMPUTING**

**SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,  
CHENNAI – 600119**

**APRIL - 2021**



# SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119

[www.sathyabamauniversity.ac.in](http://www.sathyabamauniversity.ac.in)



---

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

### **BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **SENTHIL KUMAR.B (Reg. No. 37110694)** who carried out the project entitled “**FRUITS IMAGE DETECTION USING CNN**” under my supervision from November 2020 to April 2021.

#### **Internal Guide**

Dr.V.ULAGAMUTHALVI, M.E., PhD.,

#### **Head of the Department**

Dr. S. VIGNESHWARI, M.E., Ph.D.,

Dr.L.LAKSHMANAN, M.E., Ph.D.,

---

Submitted for Viva voce Examination held on \_\_\_\_\_

Internal Examiner

External Examiner

## **DECLARATION**

I, **SENTHIL KUMAR.B (Reg.No.37110694)** hereby declare that the Project Report on **“FRUITS IMAGE DETECTION USING CNN”** is done by me under the guidance of **Dr.V.Ulagamuthalvi, M.E., Ph.D.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted inpartial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

**DATE:**

**PLACE:**

**SIGNATURE OF THE CANDIDATE**

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr.S. Vigneshwari M.E., Ph.D.,** and **Dr.L.Lakshmanan M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide, **Dr.V.Ulagamuthalvi, M.E., Ph.D.,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

## **ABSTRACT**

Fruits are very important in day-to-day life. To be healthy we need to take some fruits based on the vitamins. Nowadays people are now giving less importance to Fruits. So, we have created a model to detect fruits and vitamins which will be useful for this generation. The ultimate aim of this paper is to build a fast and accurate fruit detection system. The proposed project can detect some rare and unknown fruits. This helps us to gain some extra Knowledge. The project is mainly useful for fruit Industries to allocate the fruit in a particular box. It is an offline application. Fruit detection is used to reduce the effort of humans. To identify the fruit types used on artificial neural networks. It is the best model for the deep learning technique to easily identify the fruit types. Deep Learning models, or to be more specific, TensorFlow uses Convolutional Neural Networks (CNN) models are proposed in this project. A comparison is done between proposed algorithms and current algorithms reveals that the accuracy of fruits detection using CNN is greater than other algorithms. The dataset is then trained by manual, Alexnet, Lenet, and comparing the accuracy which has higher and it is converted into hierarchal data(.h5). It is predicted that the success obtained results will increase if the CNN is added with extra feature extraction methods to detect fruits type successfully. Thus higher accuracy is achieved and able to detect fruits and vitamins present on that fruits and deploying the model in a web application using Django Framework.

## TABLE OF CONTENTS

| CHAPTER NO. | TITLE                            | PAGE NO. |
|-------------|----------------------------------|----------|
|             | ABSTRACT                         | v        |
|             | LIST OF ABBREVIATION             | viii     |
|             | LIST OF FIGURES                  | ix       |
| 1.          | INTRODUCTION                     | 1        |
|             | 1.1 GENERAL INTRODUCTION         | 1        |
|             | DEEP LEARNING                    | 2        |
|             | CONVOLUTIONAL NEURAL NETWORK     | 4        |
|             | CNN ARCHITECTURE                 | 5        |
| 2.          | AIM,SCOPE AND LITERATURE SURVEY  | 8        |
|             | AIM                              | 8        |
|             | OBJECTIVE                        | 8        |
|             | SCOPE                            | 8        |
|             | LITERATURE SURVEY                | 9        |
| 3.          | METHODOLOGY                      | 13       |
|             | EXISTING SYSTEM                  | 13       |
|             | ARCHITECTURE DIAGRAM             | 14       |
|             | PROPOSED SYSTEM                  | 15       |
|             | UML DIAGRAMS                     | 16       |
|             | USE-CASE DIAGRAM                 | 16       |
|             | CLASS DIAGRAM                    | 17       |
|             | ACTIVITY DIAGRAM                 | 18       |
|             | SEQUENCE DIAGRAM                 | 19       |
|             | COLLABORATION DIAGRAM            | 20       |
|             | ER DIAGRAM                       | 21       |
|             | ALGORITHM                        | 22       |
|             | CREATION OF DEEP LEARNING MODELS | 24       |
|             | MODULE-1(MANUAL CNN)             | 25       |
|             | MODULE-2(ALEXNET)                | 27       |
|             | MODULE-3(LENET)                  | 29       |

|           |                                   |           |
|-----------|-----------------------------------|-----------|
|           | DJANGO FRAMEWORK                  | 31        |
|           | REQUIREMENT SPECIFICATION         | 34        |
| <b>4.</b> | <b>RESULTS AND DISCUSSION</b>     | <b>35</b> |
| <b>5.</b> | <b>CONCLUSION AND FUTURE WORK</b> | <b>40</b> |
|           | <b>REFERENCES</b>                 | <b>41</b> |
|           | <b>APPENDIX</b>                   |           |
|           | A. SAMPLE CODE                    | 43        |



## LIST OF ABBREVIATIONS

| ABBREVIATION    | EXPANSION                   |
|-----------------|-----------------------------|
| CNN             | Convolutional Neural        |
| Network HTML    | - Hypertext Markup Language |
| CSS             | Cascading Style Sheet       |
| ERD             | Entity Relationship         |
| Diagram ReLU    | - Rectified Linear          |
| Unit            |                             |
| MVT             | Model View Template         |
| IDE             | Integrated Development      |
| Environment GUI | - Graphical User Interface  |

## LIST OF FIGURES

| FIGURE NO. | FIGURE NAME                  | PAGE NO. |
|------------|------------------------------|----------|
| 1.1        | PROCESS OF DEEP LEARNING     | 3        |
| 1.2        | CNN ARCHITECTURE             | 6        |
| 3.1        | ARCHITECTURE DIAGRAM         | 14       |
| 3.2        | USE-CASE DIAGRAM             | 16       |
| 3.3        | CLASS DIAGRAM                | 17       |
| 3.4        | ACTIVITY DIAGRAM             | 18       |
| 3.5        | SEQUENCE DIAGRAM             | 19       |
| 3.6        | COLLABORATION DIAGRAM        | 20       |
| 3.7        | ER DIAGRAM                   | 21       |
| 3.8        | DEEP LEARNING                | 23       |
| 3.9        | CONVOLUTIONAL NEURAL NETWORK | 25       |
| 3.10       | ACCURACY AND LOSS GRAPH      | 26       |
| 3.11       | ALEXNET ARCHITECTURE         | 27       |
| 3.12       | ACCURACY AND LOSS GRAPH      | 29       |
| 3.13       | LENET ARCHITECTURE           | 30       |
| 3.14       | LENET LAYERS                 | 30       |
| 3.15       | DJANGO FRAMEWORK             | 32       |
| 4.1        | ANACONDA NAVIGATOR           | 35       |
| 4.2        | LENET MODEL                  | 36       |
| 4.3        | PYCHARM SCREEN               | 37       |
| 4.4        | OUTPUT AS WEB APPLICATION    | 38       |
| 4.5        | DISPLAYING IMAGE SCREEN      | 38       |
| 4.6        | FINAL OUTPUT                 | 39       |

# **CHAPTER 1**

## **INTRODUCTION**

### **GENERAL INTRODUCTION**

India is an agriculture country. Different types of fruits and vegetables are produced in India. India is at second number after china in production fruits. In India all the preharvest and post-harvest process are done manually with help of labor. Manual process is very time consuming, less efficient so to get accurate result automation in agriculture industry is needed. The post- harvest process includes sorting and grading of fruits. Different quality factors are considered for sorting and grading of fruits. These factors are internal quality factors and external quality factors. Automation is playing important role in day to day life. In India more than half population depends upon agriculture. Their main source of income is agriculture. Exporting of fresh fruit is increased day to day from India. People are very conscious about their health; they prefer only fresh, good quality fruit. Texture, Color and Size are the important parameters for fruit quality identification. The color recognition is very important process in ripeness detection. The ripeness detection is external quality factor. But texture is also very important. Because of texture defected fruit can be recognized. Texture analysis detects the non-uniformity of fruit outer surface. The size is also important parameter. It clearly seen parameter all customer select fruit based on size.

The ultimate aim of this paper is to propose a dataset of images containing popular fruits. The High quality of the dataset is collected and split into train and test images. The convolutional Neural Network plays a vital role in training the model using Tensorflow and Keras Library. Both Libraries are free and open-source which provides an interface for Artificial Neural Network. Deep learning is also known as structured learning it is part of machine learning.

In Deep Learning, a convolutional neural network(CNN/ConvNet) it is a class of deep neural network, mostly applied for visual imaginary. Learning can be supervised,semi-supervised, or unsupervised.

For the second objective, we have trained a Convolutional Neural Network Model which is capable of identifying fruits and vitamins. This model will be useful

in sorting the fruits in current companies working in the Augmented Reality field. Google Lens is online we need network connectivity to search but in our project, we don't need internet and it's an offline application that will be user-friendly. The accuracy is high compared to other machine learning algorithms. The model is saved into Hierarchical data and deployed as a web application using Django Framework with Sqlite3 database.

Recognizing different kinds of fruits is a difficult task in supermarkets, since the cashier must point out the categories of a particular fruit to determine its price. The use of barcodes has mostly ended this problem for packaged products but given that most consumers want to pick their products, they cannot be pre-packaged, and thus must be weighed. A solution is issuing codes for every fruit, but the memorization is problematic leading to pricing errors. Another solution is to issue the cashier an inventory with pictures and codes, however, flipping over the booklet is time consuming. Automatic classification of fruits via computer vision is still a complicated task due to the various properties of many types of fruits. The fruit image detection technique which was based on external properties of fruits such as shape, size and color.

## **DEEP LEARNING**

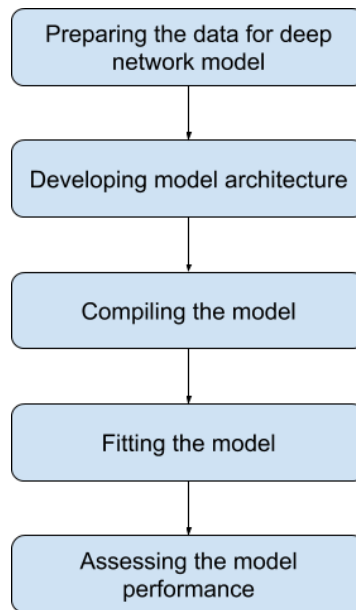
Deep Learning is a field which comes under Machine Learning and is related to the use of algorithms in artificial neural networks. It is majorly used to create a predictive model to solve the problems with just a few lines of coding. A Deep Learning system is an extensive neural network which is inspired by the function and structure of the brain. Deep Learning is essential, especially when vast amounts of data are involved.

It creates an extensive neural network, and with the help of a large number of data, it becomes scalable and in return, improves the performance. It is beneficial especially in the case of unstructured data or the data which are unlabeled. Deep Learning can give excellent results through supervised learning or learning from labelled data.

As there are lots of data available on the internet which are generated every day

and where the majority of them are unstructured, Deep Learning is becoming the next big thing in solving and dealing with these kinds of problems. While in a

situation where massive data becomes a problem to process and analyze, on the other hand, deep learning becomes better and better with more data given to it. It creates a bigger and better neural network when more data are connected in many ways creating bigger models and more computations processing. It also provides scope for better and improved algorithms, new insights, and enhanced techniques.



**Fig 1.1: Process of Deep Learning**

In the last few years of the IT industry, there has been a huge demand for once particular skill set known as Deep Learning. Deep Learning a subset of Machine Learning which consists of algorithms that are inspired by the functioning of the human brain or the neural networks.

These structures are called as Neural Networks. It teaches the computer to do what naturally comes to humans. Deep learning, there are several types of models such as the Artificial Neural Networks (ANN), Autoencoders, Recurrent Neural Networks (RNN) and Reinforcement Learning. But there has been one particular model that has contributed a lot in the field of computer vision and image analysis which is the Convolutional Neural Networks (CNN) or the ConvNets.

CNNs are a class of Deep Neural Networks that can recognize and classify particular features from images and are widely used for analyzing visual images. Their applications range from image and video recognition, image classification, medical image analysis, computer vision and natural language processing.

The term 'Convolution' in CNN denotes the mathematical function of convolution which is a special kind of linear operation wherein two functions are multiplied to produce a third function which expresses how the shape of one function is modified by the other. In simple terms, two images which can be represented as matrices are multiplied to give an output that is used to extract features from the image.

## **CONVOLUTIONAL NEURAL NETWORK**

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on the shared-weight architecture of the convolution kernels that scan the hidden layers and translation invariance characteristics. They have applications in image and video recognition, recommender systems, image classification and Image segmentation, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.

CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include varying the weights as the loss function gets minimized while randomly trimming connectivity. CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in the filters. Therefore, on the scale of connectedness and complexity, CNNs are on the lower extreme.

Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive

fields of different neurons partially overlap such that they cover the entire visual field.

CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns to optimize the filters or convolution kernels that in traditional algorithms are hand-engineered. This independence from prior knowledge and human intervention in feature extraction is a major advantage.

Convolutional neural networks (CNNs) are also made up of neurons, that have learnable weights and biases. That's why in this manner, they are like ordinary neural networks (NNs).

If we recall the working of ordinary NNs, every neuron receives one or more inputs, takes a weighted sum and it passed through an activation function to produce the final output. Here, the question arises that if CNNs and ordinary NNs have so many similarities then what makes these two networks different to each other?

What makes them different is the treatment of input data and types of layers? The structure of input data is ignored in ordinary NN and all the data is converted into 1-D array before feeding it into the network.

But, Convolutional Neural Network architecture can consider the 2D structure of the images, process them and allow it to extract the properties that are specific to images. Moreover, CNNs have the advantage of having one or more Convolutional layers and pooling layer, which are the main building blocks of CNNs.

These layers are followed by one or more fully connected layers as in standard multilayer NNs. So, we can think of CNN, as a special case of fully connected networks.

### ***Convolutional Neural Network (CNN) architecture:***

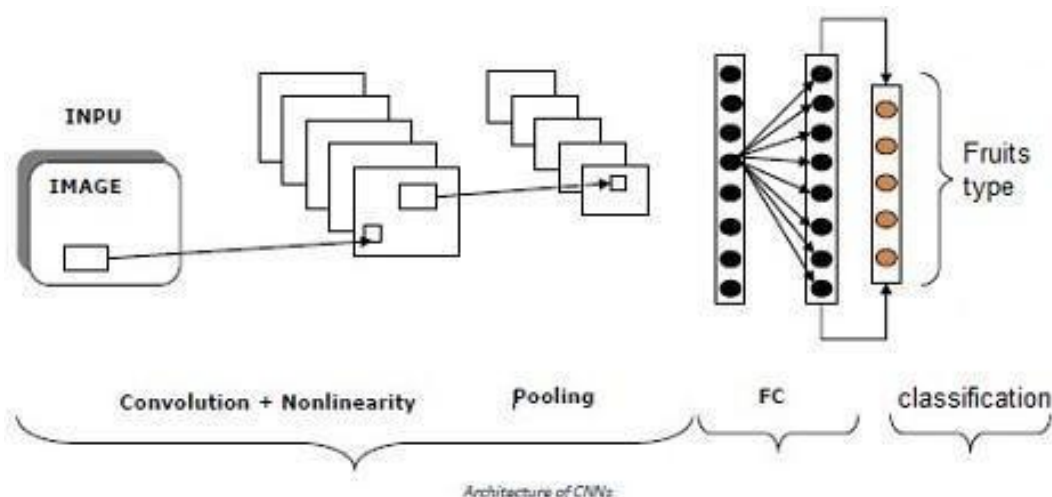
The architecture of CNN is basically a list of layers that transforms the 3-dimensional, i.e. width, height and depth of image volume into a 3-dimensional output volume. One important point to note here is that, every neuron in the current layer is connected to a small patch of the output from the previous layer,

which is like overlaying a  $N \times N$  filter on the input image.

It uses  $M$  filters, which are basically feature extractors that extract features like edges, corner and so on. Following are the layers **[INPUT-CONV-RELU-POOL-FC]** that are used to construct Convolutional neural networks (CNNs)–

- **INPUT**– As the name implies, this layer holds the raw pixel values. Raw pixel values mean the data of the image as it is. Example, INPUT [64×64×3] is a 3-channelled RGB image of width-64, height-64 and depth-3.
- **CONV**– This layer is one of the building blocks of CNNs as most of the computation is done in this layer. Example - if we use 6 filters on the above mentioned INPUT [64×64×3], this may result in the volume [64×64×6].
- **RELU**–Also called rectified linear unit layer, that applies an activation function to the output of previous layer. In other manner, a non-linearity would be added to the network by RELU.
- **POOL**– This layer, i.e. Pooling layer is one other building block of CNNs. The main task of this layer is down-sampling, which means it operates independently on every slice of the input and resizes it spatially.
- **FC**– It is called Fully Connected layer or more specifically the output layer. It is used to compute output class score and the resulting output is volume of the size 1×1×L where L is the number corresponding to class score.

The diagram below represents the typical architecture of CNNs–



**Fig 1.2 : Convolutional Neural Network Architecture**

## CREATING CNN STRUCTURE

We have seen the architecture and the basics of CNN, now we are going to building convolutional network using CNTK. Here, we will first see how to put together the structure of the CNN and then we will look at how to train the parameters of it.

At last we'll see, how we can improve the neural network by changing its structure with various different layer setups. We are going to use MNIST image dataset.

So, first let's create a CNN structure. Generally, when we build a CNN for recognizing patterns in images, we do the following–

- We use a combination of convolution and pooling layers.
- One or more hidden layer at the end of the network.

At last, we finish the network with a softmax layer for classification purpose

The Jupyter Notebook is an incredibly powerful tool for interactively

developing and presenting data science projects. This article will walk you through how to set up Jupyter Notebooks on your local machine and how to start using it to do data science projects.

First, though: what is a “notebook”? A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media. This intuitive workflow promotes iterative and rapid development, making notebooks an increasingly popular choice at the heart of contemporary data science, analysis, and increasingly science at large. Best of all, as part of the open source Project Jupyter, they are completely free. The Jupyter project is the successor to the earlier IPython Notebook, which was first published as a prototype in 2010. Although it is possible to use many different programming languages within Jupyter Notebooks, this article will focus on Python as it is the most common use case. (Among R users, R Studio tends to be a more popular choice).

To get the most out of this tutorial you should be familiar with programming,

specifically Python and pandas specifically. Jupyter Notebooks can also act as a flexible platform for getting to grips with pandas and even Python, as will become apparent in this article.

## **AIM OF PROJECT ER2**

**C  
HA  
PT  
  
AI  
M  
AN  
D  
SC  
OP  
E**

Now a days many people are giving less important to fruit. Some people might not knowing how much vitamins present in it. So this model will detect the fruits with vitamins in it. So that people can select the fruits which is beneficial for them. The ultimate aim of this project is to detect the real time fruits with vitamins present in it. The fruits are very important in day-to-day life, and has lot of vitamins present in every fruit. The proposed model able to detect the fruits and vitamins accurately without any error and we have created a web application using the model and deployed in Django Framework using Pycharm IDE which is specifically for python.

### ***OBJECTIVES***

The proposed project is able to recognize the fruit based on the features like shape, colour, and texture. This increases the knowledge of common people about some rare and unknown fruits. The project is mainly concentrating on reducing human effort and making human life easier. To identify the fruits types used on artificial neural network. It is best model for deep learning technique to easily identify the fruits types.

### **SCOPE**

We have trained a Convolutional Neural Network Model which is capable of identifying fruits and vitamins. This model will be useful in sorting the fruits in current companies working in the Augmented Reality field. Google Lens is online we need network connectivity to search but in our project, we don't need internet and it's an offline application that will be user-friendly.

#### Advantages:

The proposed project is able to detect the fruits type and vitamins.

This helps us to gain some extra knowledge.

Fruit detection will be able to reduce the current ongoing problems. It reduces confusion among the particular fruit .  
Its an offline application.

## **LITERATURE SURVEY**

In [1] “Machine Learning Driven Approach Towards the Quality Assessment of Fresh Fruits Using Non-invasive Sensing” by Aifeng Ren, Adnan Zahid, Ahmed Zoha, Syed Aziz Shah States that in agriculture science, accurate information of moisture content (MC) in fruits and vegetables in an automated fashion can be vital for astute quality and grading evaluation. This demands for a viable, feasible and cost-effective technique for the defect recognition using timely detection of MC in fruits and vegetables to maintain a healthy sensory characteristic of fruits. They propose a non-invasive machine learning (ML) driven technique to monitor variations of MC in fruits using the terahertz (THz) waves with Swissto12 material characterization kit (MCK) in the frequency range of 0.75 THz to 1.1 THz. In this regard, multi-domain features are extracted from time-frequency, and time-frequency domains, and applied three ML algorithms such as support vector machine (SVM), knearest neighbour (KNN) and Decision Tree (D-Tree) for the precise assessment of MC in both apple and mango slices. The results illustrated that the performance of SVM exceeded other classifiers results using 10-fold validation and leave-oneobservation-out-cross-validation techniques. The novel and non-invasive sensing technique for the quality assessment of fresh of fruits by integrating machine learning ML with terahertz (THz) waves. For this purpose, scattering measurements of both apple and mango slices were obtained using Swissto12 MCK system. Multiple domain features such as time-, frequency-, and time-frequency domains were extracted and the classification was performed to determine the moisture content (MC) in fruits slices using SVM, KNN and D-tree classifiers. Results showed that by discarding unwanted features and applying comprehensive cross-validation technique, classification accuracy was substantially improved which eventually helped to reduce the computational time. Furthermore, it was perceived that, in most of the cases, the SVM classifier based

on RBF kernel outperformed KNN and D-Tree classifier for both fruit slices. Thus, SVM

demonstrated more precise quality assessments of fresh fruits by determining their MC more precisely for four consecutive days.

In [2] “Weakly Supervised Fruit Counting for Yield Estimation Using Spatial Consistency” was proposed by Enrico Bellocchio, Thomas A. Ciarfuglia, Gabriele Costante states that Fruit counting is a fundamental component for yield estimation applications. Most of the existing approaches address this problem by relying on fruit models. Despite the impressive results achieved by these approaches, all of them need strong supervision information during the training phase. In agricultural applications, manual labeling may require a huge effort or, in some cases, it could be impossible to acquire fine-grained ground truth labels. In this letter, we tackle this problem by proposing a weakly supervised framework that learns to count fruits without the need for task-specific supervision labels. a novel weakly-supervised framework for fruit counting in agricultural applications. The WSCOUNT strategy is able to learn to count without requiring task-specific supervision labels, such as manually labelled object bounding boxes or total instance count. Instead, it exploits a simple binary presence absence classifier (PAC), trained with only image level labels, and a spatial consistency loss that imposes coherency among counting branches at different scale levels. By removing the necessity for laborious data collection processes, our approach takes an important step towards the complete automation of yield estimation systems. The experiments run on three different fruit species clearly show that our approach guarantees performances that are comparable to those of fully supervised baselines.

In [7] “Classification of Cape Gooseberry Fruit According to Its Level of Ripeness Using Machine Learning Techniques and Different Color Spaces” A Survey proposed by wilson castro , jimy oblitas , miguel de-la-torre The classification of fresh fruits according to their visual ripeness is typically a subjective and tedious task; consequently, there is a growing interest in the use of non-contact techniques to automate this process. Machine learning techniques, such as artificial neural

networks, support vector machines (SVMs), decision trees, and K-nearest neighbor algorithms, have been successfully applied for classification problems in the literature, particularly for images of fruit. However, the particularities of each

classification problem make it difficult, if not impossible, to select a general technique that is applicable to all types of fruit. In this paper, the combinations of four machine learning techniques and three color spaces (RGB, HSV, and  $L^*a^*b^*$ ) were evaluated with regard to their ability to classify Cape gooseberry fruits. To this end, 925 Cape gooseberry fruit samples were collected, and each fruit was manually classified into one of seven different classes according to its level of ripeness. Develop a non-intrusive system for the classification of gooseberry fruits according to their degree of ripeness. Twelve classification models were developed by combining four machine learning techniques (the ANN, KNN, DT, and SVM classifiers) with three color spaces (RGB, HSV, and  $L^*a^*b^*$ ). The color space utilized for the classification clearly influenced the accuracy of the sorting system; this dynamic was observed mainly in the models based on the ANN and SVM techniques. Meanwhile, the models based on the KNN and DT techniques yielded good results regardless of the color space used. The  $L^*a^*b^*$  color space produced good results regardless of the machine learning technique employed. However, the classifier developed from the combination of the SVM technique and  $L^*a^*b^*$  color space gave the best performance in terms of the accuracy and f-measure.

In [6] "A SAR Image Classification Algorithm Based on Multi-Feature Polarimetric Parameters Using FOA and LS-SVM" A Survey done by shiyu lu, kamal sarabandi, Ieland pierce have done research on Synthetic Aperture Radar (SAR) image classification algorithm based on multi-feature using Fruit Fly Optimization Algorithm (FOA) and Least Square Support Vector Machine (LS-SVM). First, pixel-based information derived from three elements of coherency matrix, six parameters obtained by  $H/\alpha/A$  decomposition and Freeman decomposition techniques, and three polarimetric parameters including the total receive power (SPAN), pedestal height, and Radar Vegetation Index (RVI), as well as region-based information derived from eight texture parameters obtained by Grey Level Co-occurrence Matrix (GLCM) are combined to use as the features of land cover. Second, Kernel Principal Component Analysis (KPCA) is used to reduce the

dimensionality of the multi- feature data derived from the integration of the pixel-based and region-based information. Third, LS-SVM is used as the classifier in this study due to its fast solving speed and desirable classification capability. Since the input parameters of

LS-SVM significantly affect the classification performance, we employ FOA to obtain the optimized input parameters. a classification method based on multi-feature using FOA and LS-SVM. The basic features for a SAR image are polarimetric parameters. Coherency matrix and six parameters obtained by  $H/\alpha/A$  decomposition and Freeman decomposition, as well as the total receive power (SPAN), polarimetric pedestal height, and (RVI) as the features of land cover. In addition to pixel-based information from polarimetric parameters, region-based information is another significant feature for an image. GLCM is an efficient tool to obtain region-based information. Thus, eight representative textural parameters obtained by GLCM, which are CON, ASM, COR, ENT, IDM, MAX, DIS, and INV are introduced to obtain texture information. However, while the integration of multi-feature characterizes the ground object better, it results in the curse of dimensionality and feature redundancy.

In [5] "Classification Of Eligibility Consumption Of Manalagi Apple Fruit Varieties Using Backpropagation" A survey done by Muladi Muladi, Dyah Lestari, Dedy Tri Prasetyo states that the quality of Manalagi apple fruit varieties prior to sorting them automatically. The sorting system used is still manually in fruit sorting with the help of human labor that takes a long time. The sorting process is slow therefore the sorted result is low in number and inconsistent in quality. In this study, the classification system is automatically developed for the Manalagi apple fruit with the extraction of the average RGB color feature with the Backpropagation algorithm. The purpose of the developed system is to distinguish the Manalagi apple fruit of worth consumption and not worth the consumption. The fruit classification is based on the average RGB color composition of the fruit skin. Systems developed using image processing such as image segmentation, color feature extraction with average RGB models and fruit quality classification decisions with the Backpropagation algorithm. The classification of Manalagi apple fruit using image processing and neural network based on the value of input fruit skin color. Changes in the skin of apple fruit that daily change the level of brightness become important in the classification process. Color changes in the

skin of Manalagi apple fruit from the harvest of trees to begin to rot every fruit varies. The characteristic of Manalagi apple fruit skin worth consumption has RGB value Green color is more dominant.

**C**

**H**

**A**

**EXISTING SYSTEM:**

### **PTER3 METHODOLOGY**

This reports four-beam 1 GHz/2.3 GHz dual frequency sensor system to assess the quality of thick rind pomelo fruits. The principle of the proposed sensor system is based on phase difference between lower- and higher-frequency reflection coefficients. The sensor system consists of five dual-band antennas and a customized circuit to approximate. The main components of the customized circuit are down conversion mixers and phase detector module. Growing global consumption of thick rind fruits has contributed to their greater economic significance. There are a variety of thick rind fruits, such as pomelo, watermelon,

and durian. To command premium prices, quality control of the fruits is paramount. Conventionally, fruit quality assessment relies on human sensory input. However, the practice is subjective, leading to inconsistent fruit quality and reputation damage. Thick rind fruits, e.g., pomelo, watermelon, are typically of spherical shape or, specifically, concentric dielectric sphere with rind as the outer layer and flesh as the inner layer. In this research, simulations were carried out using a pomelo fruit model whose outer and inner diameters are 16 and 12 cm, which are the typical size of pomelo fruit. The flesh portion of a typical pomelo fruit consists of 15 sections, and the simulations are to differentiate between normal and granulated pomelo fruits. Granulation is a condition in which juice vesicles become dry and tough, which makes the fruit unpalatable. In the simulation, the pomelo that contains one granulated section (out of 15 sections) is regarded as failing. Prior to simulation, the relative permittivity of real pomelo fruits (three each for normal and granulated pomelos) were determined using Agilent 85070E dielectric probe connected to N9916A Field Fox microwave vector network analyzer (Figure 3(a)). The relative permittivity of the rind and flesh was measured using the dielectric probe, To obtain accurate relative permittivity, the pomelo flesh was homogeneously pressed prior to measurement by the dielectric probe. They have Simulations were carried out using single- (1 GHz) and dual-frequency (1 GHz/2.3 GHz) sensor systems; and the results indicated that the single-frequency scheme achieved lower accuracy in classification of pomelo fruits. As a result, a prototype of four beam dual-frequency

sensor system was fabricated, and experiments were carried out using plastic pomelo models and real pomelo fruits. They further demonstrate The experimental results showed that the average of the normal and granulated pomelo fruits are almost identical, while SD of the granulated pomelo is noticeably larger than that of the normal pomelo. As such, SD is used as the determinant of pomelo quality.

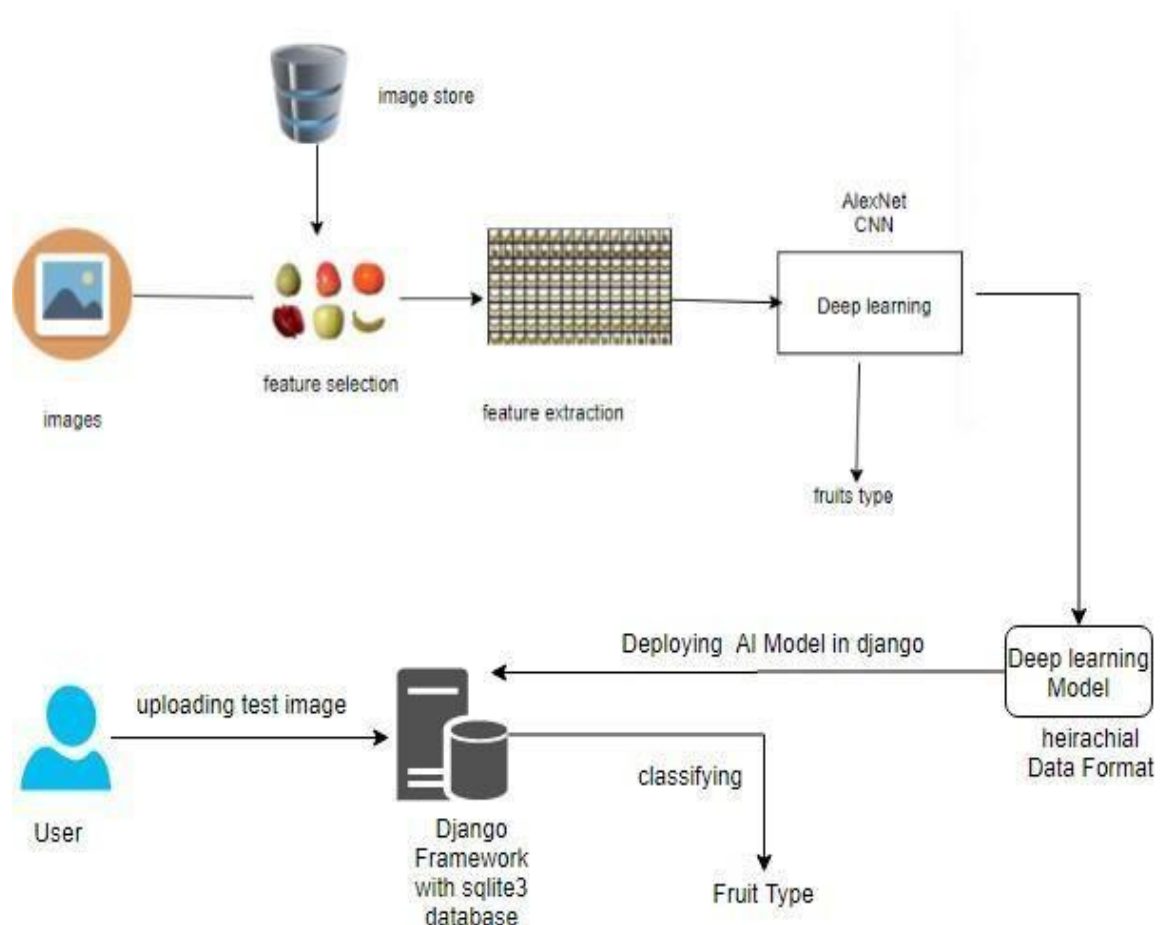
#### **Drawback:**

It has not focused on identifying Alex Net CNN as classifier.

They are not using deep learning technique

It has not focused on increasing the recognition rate and classification accuracy of severity of fruits types.

#### **ARCHITECTURE DIAGRAM**



**Fig 3.1 : Architecture Diagram**

The architecture above Fig 3.1 shows the working model of the project. First, we train the images from the dataset and the deep learning model is created and comparing the accuracies of the model. The model which has high accuracy is converted into a hierarchical data format and deploying the model in Django Framework as a desktop application.

### **PROPOSED SYSTEM:**

The proposes a new and robust machine learning model based on a convolutional neural network (CNN) to automatically classify single cells in thin blood smears on standard microscope slides as either infected or uninfected. there is a trade-off to make: will we create a model that highly specializes in variety, or will we create a generic model that can be capable of recognizing some fruits.

The proposed algorithms can automatically extract complex image fruits types and successfully detect fruits in different scenes. as input, we used whole images, so it was not necessary to perform any preprocessing or the fruits, samples of more number of images are collected that comprised of different classes data. different number of images is collected for each classes that was classified into input images.

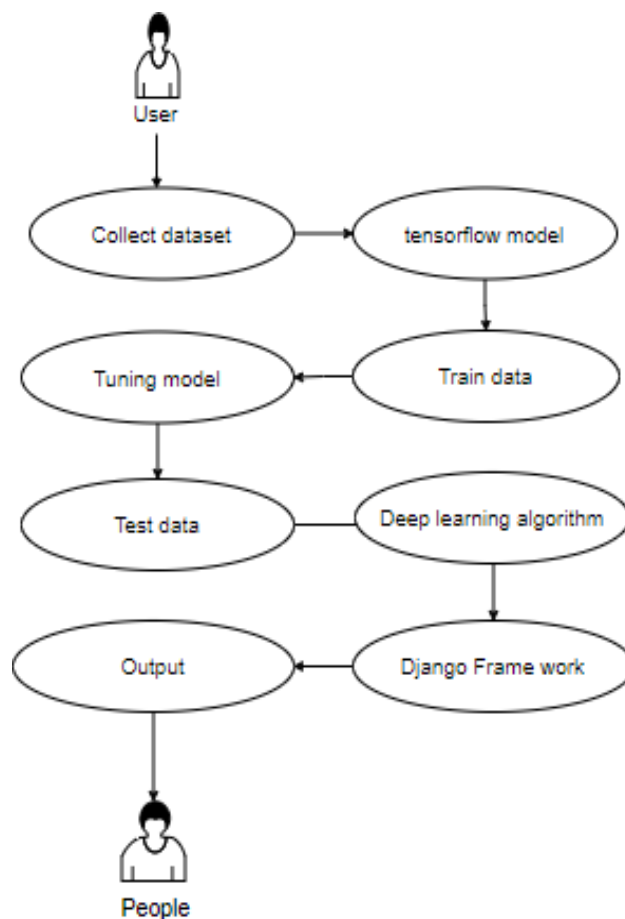
We proposed a Deep Learning (DL) based fruits types classification method to prevent fruits. The deep learning method used in the study is the LeNet Convolutional Neural Network (CNN). It is predicted that the success of the obtained results will increase if the CNN method is supported by adding extra feature extraction methods and classify successfully fruits types.

## UML DIAGRAM:

### *Use case diagram:*

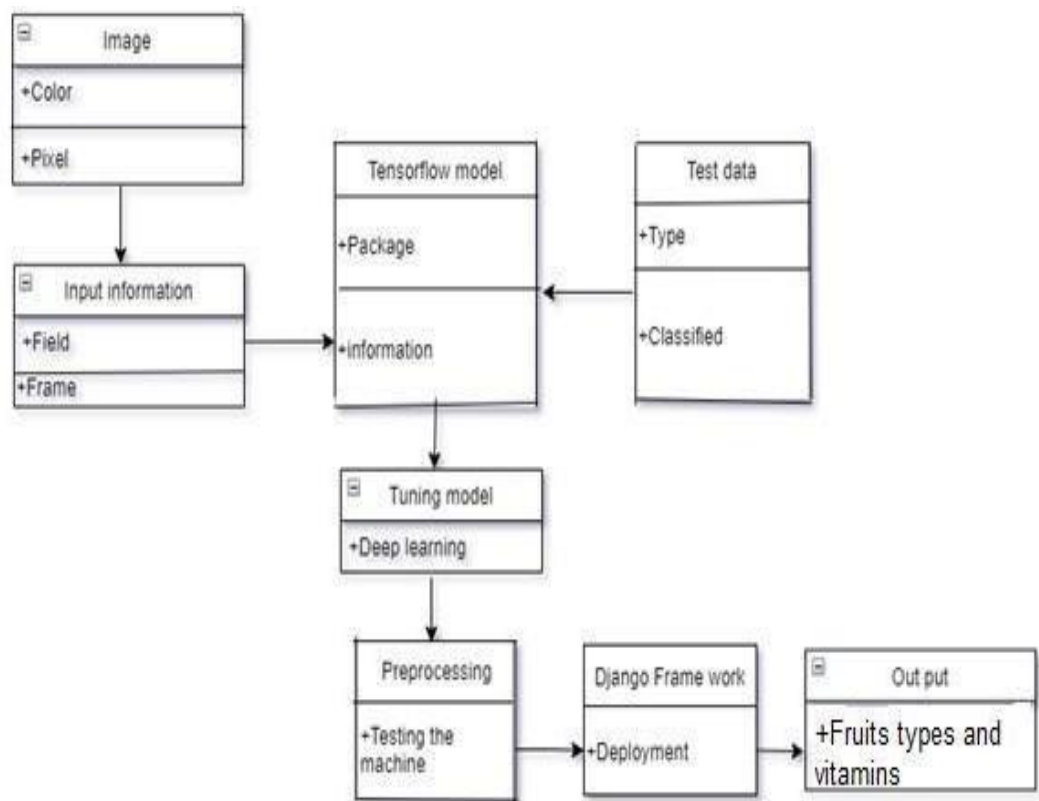
The purpose of use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (activity, sequence, collaboration, and State chart) also have the same purpose. We will look into some specific purpose, which will distinguish it from other four diagrams.

Use case diagrams are considered for high level requirement analysis of a system. So when the requirements of a system are analyzed the functionalities are captured in use cases. So, it can say that uses cases are nothing but the system functionalities written in an organized manner. Now the second things which are relevant to the use cases are the actors.



**Fig 3.2 : Use-case Diagram**

### ***Class Diagram:***

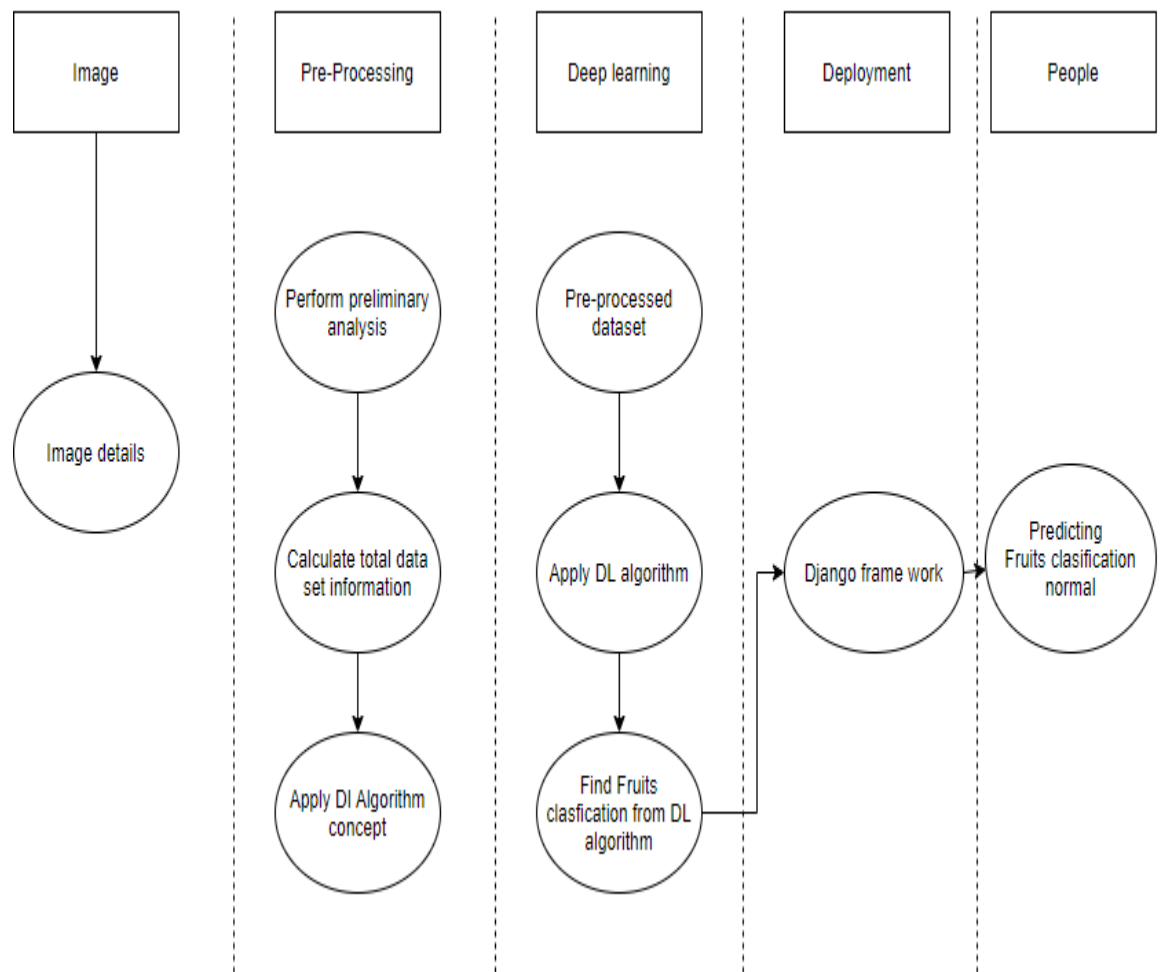


**Fig 3.3 : Class Diagram**

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So a collection of class diagrams represent the whole system. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance Responsibility (attributes and methods) of each class should be clearly identified for each class minimum number of properties should be specified and because, unnecessary properties will make the diagram complicated. Use notes whenever required to describe some aspect of the diagram and at the end of the drawing it should be understandable to the developer/coder. Finally, before making the final version, the diagram should be drawn on plain paper

and rework as many times as possible to make it correct.

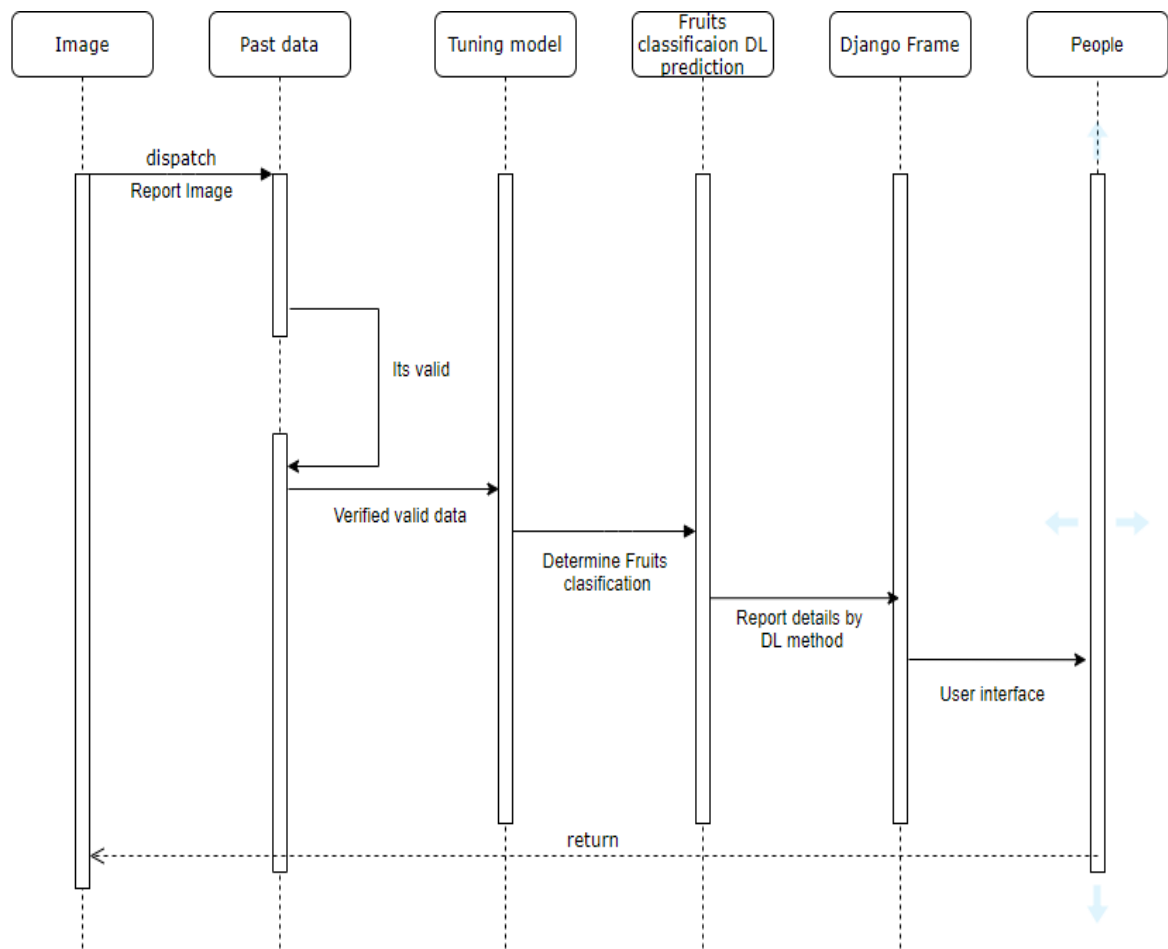
### Activity Diagram:



**Fig 3.4 : Activity Diagram**

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part. It does not show any message flow from one activity to another. Activity diagram is some time considered as the flow chart. Although the diagrams looks like a flow chart but it is not. It shows different flow like parallel, branched, concurrent and single.

### Sequence Diagram:

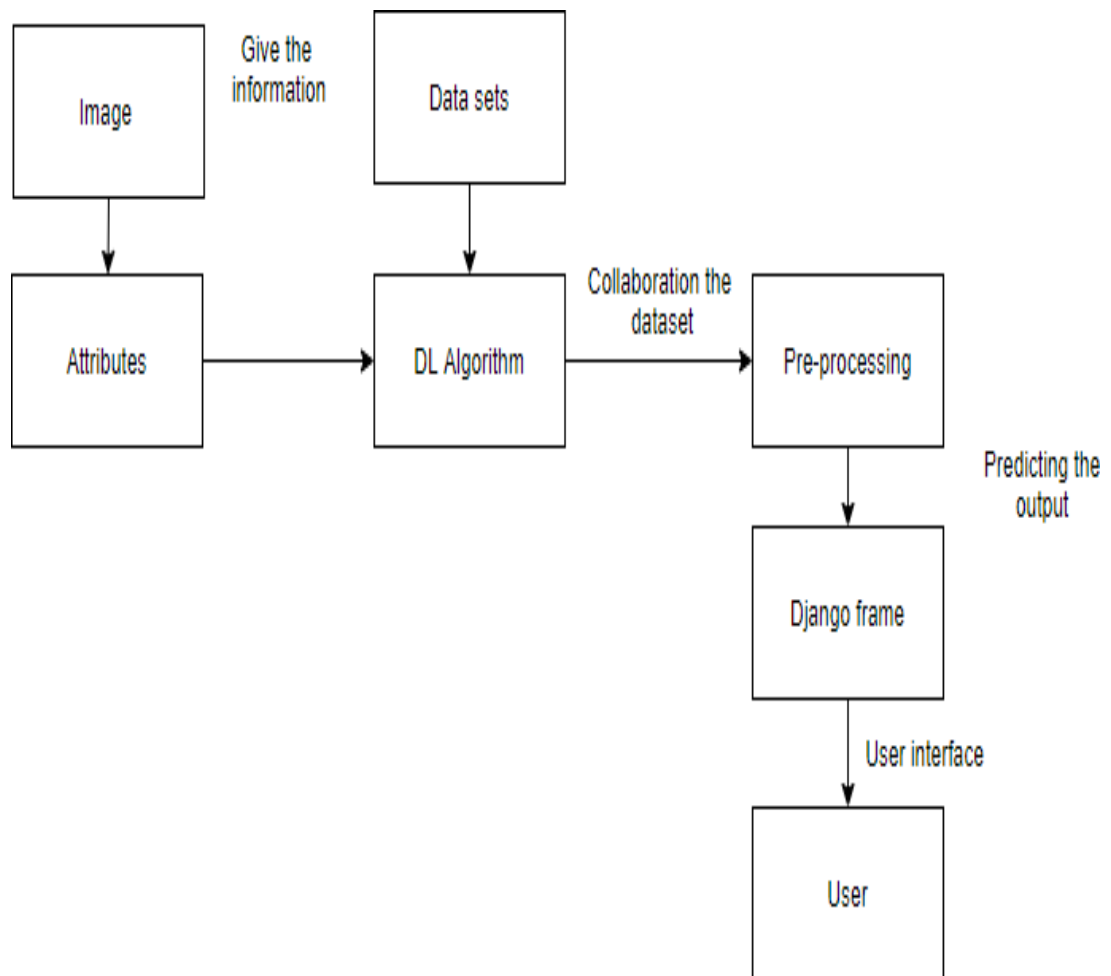


**Fig 3.5 : Sequence Diagram**

Sequence diagrams model the flow of logic within your system in a visual manner, enabling you both to document and validate your logic, and are commonly used for both analysis and design purposes. Sequence diagrams are the most popular UML artifact for dynamic modeling, which focuses on identifying the behavior within your system.

Other dynamic modeling techniques include activity diagramming, communication diagramming, timing diagramming, and interaction overview diagramming. Sequence diagrams, along with class diagrams and physical data models are in my opinion the most important design-level models for modern business application development.

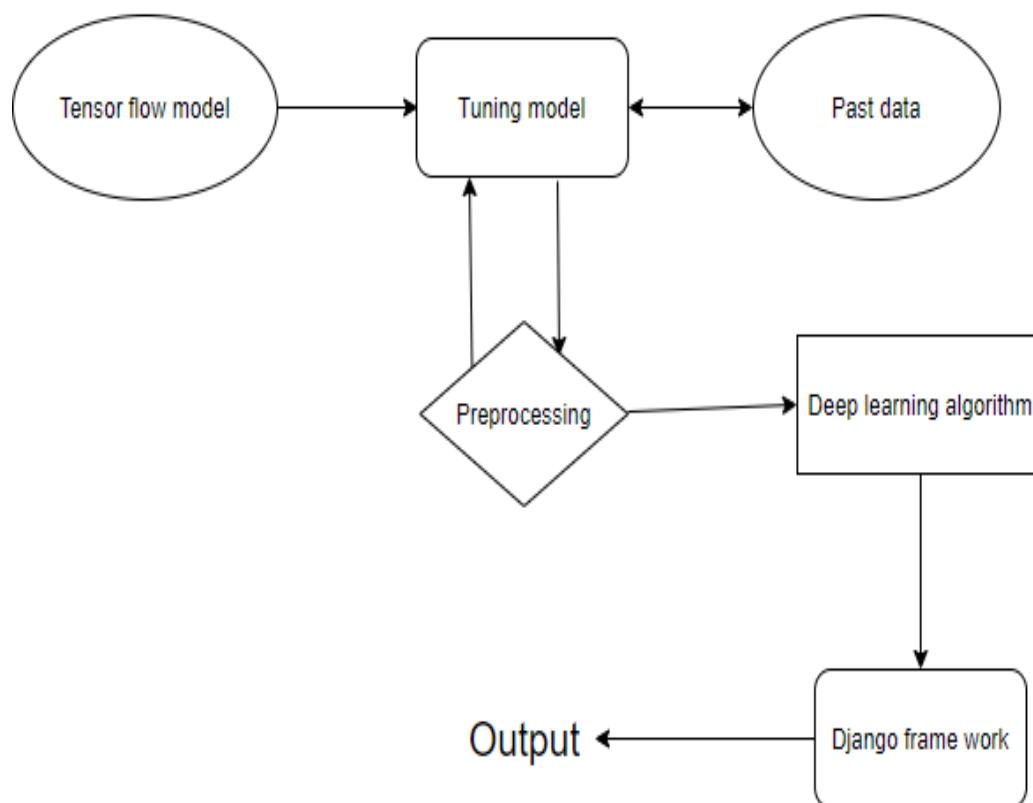
### ***Collaboration Diagram:***



**Fig 3.6 : Collaboration Diagram**

A collaboration diagram is a type of visual presentation that shows how various software objects interact with each other within an overall IT architecture and how users can benefit from this collaboration. A collaboration diagram often comes in the form of a visual chart that resembles a flow chart. It can show, at a glance, how a single piece of software complements other parts of a greater system.

**ER Diagram:**



**Fig 3.7 : Entity Relationship Diagram**

An entity relationship diagram (ERD), also known as an entity relationship model, is a graphical representation of an information system that depicts the relationships among people, objects, places, concepts or events within that system. An ERD is a data modeling technique that can help define business processes and be used as the foundation for a relational database. Entity relationship diagrams provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later.

## **ALGORITHM:**

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

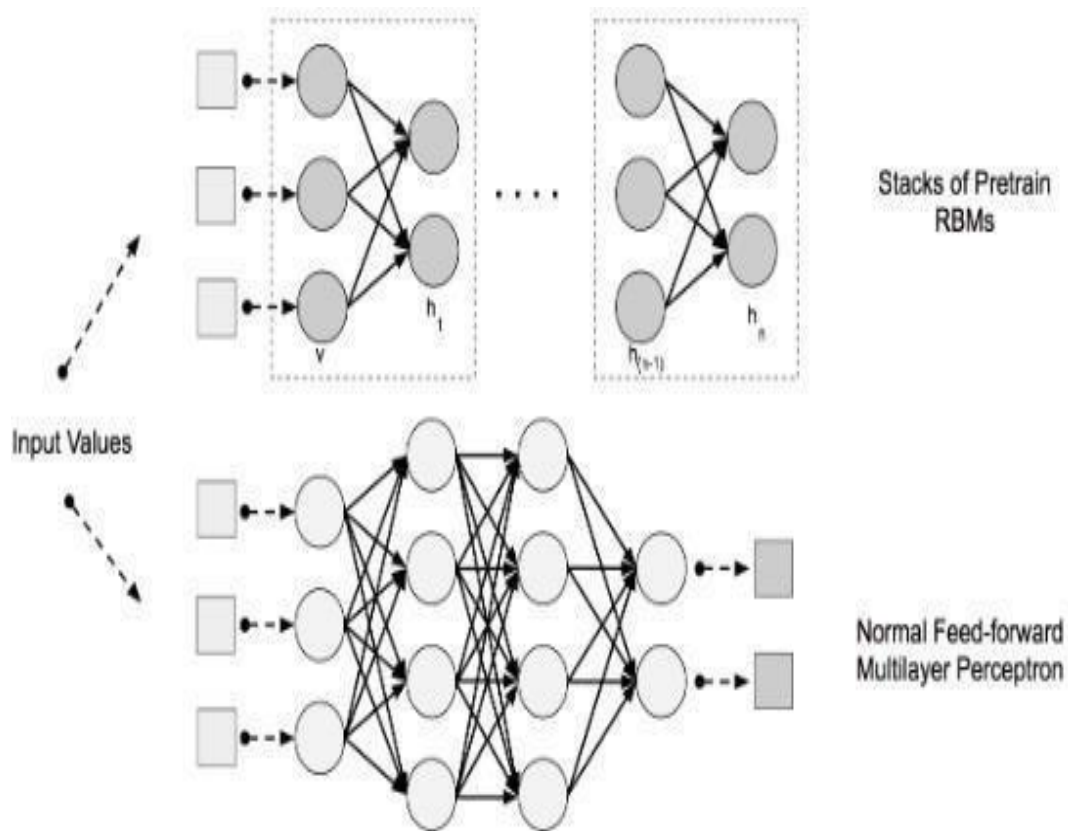
In a word, accuracy. Deep learning achieves recognition accuracy at higher levels than ever before. This helps consumer electronics meet user expectations, and it is crucial for safety-critical applications like driverless cars. Recent advances in deep learning have improved to the point where deep learning outperforms humans in some tasks like classifying objects in images.

1. Deep learning requires large amounts of labeled data. driverless car development requires millions of images and thousands of hours of video.
2. Deep learning requires substantial computing power. High-performance GPUs have a parallel architecture that is efficient for deep learning. When combined with clusters or cloud computing, this enables development teams to reduce training time for a deep learning network from weeks to hours or less.

Most deep learning methods use neural network architectures, which is why deep learning models are often referred to as deep neural networks. The term "deep" usually refers to the number of hidden layers in the neural network. Traditional

neural networks only contain 2-3 hidden layers, while deep networks can have as many as 150.

Deep learning models are trained by using large sets of labeled data and neural network architectures that learn features directly from the data without the need for manual feature extraction



**Fig 3.8 : Deep Learning**

One of the most popular types of deep neural networks is known as convolutional neural networks (CNN or ConvNet). A CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images.

CNNs eliminate the need for manual feature extraction, so you do not need to identify features used to classify images. The CNN works by extracting features

directly from images. The relevant features are not retrained; they are learned while

the network trains on a collection of images. This automated feature extraction makes deep learning models highly accurate for computer vision tasks such as object classification.

CNNs learn to detect different features of an image using tens or hundreds of hidden layers. Every hidden layer increases the complexity of the learned image features. For example, the first hidden layer could learn how to detect edges, and the last learns how to detect more complex shapes specifically catered to the shape of the object we are trying to recognize.

## **CREATION OF DEEP LEARNING MODELS**

To train a deep network from scratch, you gather a very large labeled data set and design a network architecture that will learn the features and model. This is good for new applications, or applications that will have a large number of output categories. This is a less common approach because with the large amount of data and rate of learning, these networks typically take days or weeks to train.

### **Transfer Learning**

Most deep learning applications use the transfer learning approach, a process that involves fine-tuning a pretrained model. You start with an existing network, such as AlexNet or GoogLeNet, and feed in new data containing previously unknown classes. After making some tweaks to the network, you can now perform a new task, such as categorizing only dogs or cats instead of 1000 different objects. This also has the advantage of needing much less data (processing thousands of images, rather than millions), so computation time drops to minutes or hours

### **Feature Extraction**

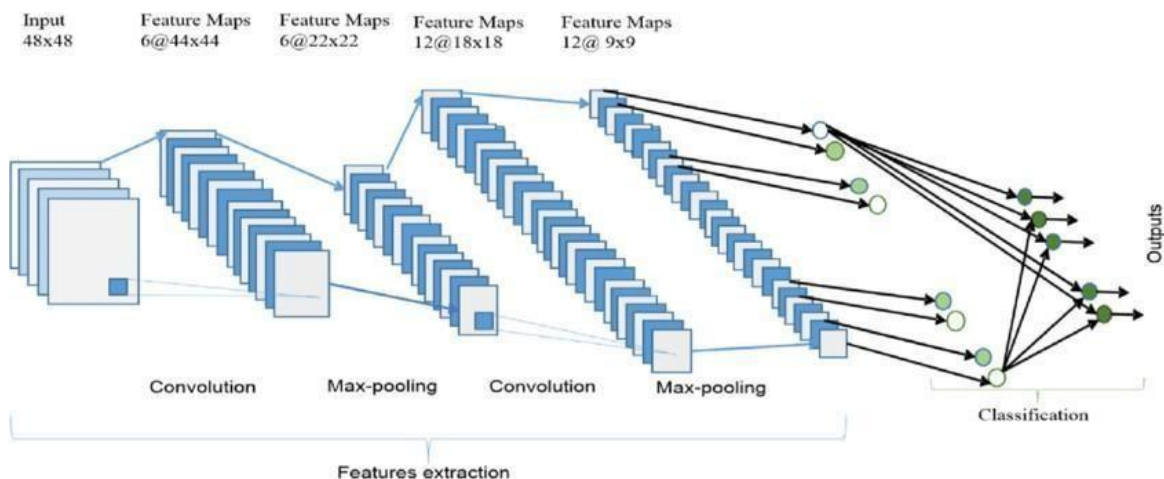
A slightly less common, more specialized approach to deep learning is to use the network as a **feature extractor**. Since all the layers are tasked with learning

certain features from images, we can pull these features out of the network at any time during the training process.

## MODULE-1 (MANUAL CNN) :

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on the shared-weight architecture of the convolution kernels that scan the hidden layers and translation invariance characteristics. They have applications in image and video recognition, recommender systems, image classification, Image segmentation, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.

CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include varying the weights as the loss function gets minimized while randomly trimming connectivity. CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble patterns of increasing complexity using smaller and simpler patterns embossed in the filters.



**Fig 3.9 : Convolution Neural Network**

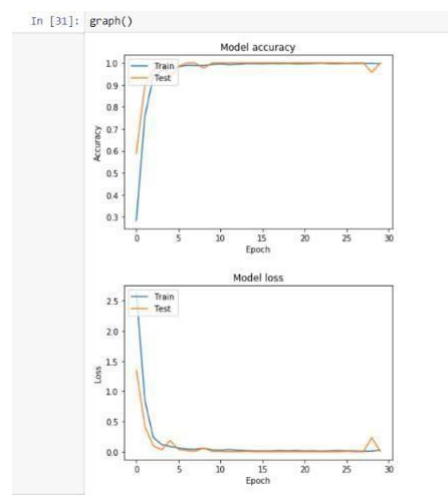
Input layer in CNN contain image data. Image data is represented by three dimensional matrixes. It needs to reshape it into a single column. Suppose you

have image of dimension  $28 \times 28 = 784$ , it need to convert it into  $784 \times 1$  before feeding into input.

Convo layer is sometimes called feature extractor layer because features of the image are get extracted within this layer. First of all, a part of image is connected to Convo layer to perform convolution operation as we saw earlier and calculating the dot product between receptive fields (it is a local region of the input image that has the same size as that of filter) and the filter. Result of the operation is single integer of the output volume. Then the filter over the next receptive field of the same input image by a Stride and do the same operation again. It will repeat the same process again and again until it goes through the whole image. The output will be the input for the next layer. Pooling layer is used to reduce the spatial volume of input image after convolution. It is used between two convolution layers. If it applies FC after Convo layer without applying pooling or max pooling, then it will be computationally expensive. So, the max pooling is only way to reduce the spatial volume of input image. It has applied max pooling in single depth slice with Stride of 2. It can observe the 4 x 4 dimension input is reducing to 2 x 2 dimensions. Fully connected layer involves weights, biases, and neurons. It connects neurons in one layer to neurons in another layer. It is used to classify images between different categories by training.

SoftMax or Logistic layer is the last layer of CNN. It resides at the end of FC layer. Logistic is used for binary classification and SoftMax is for multi-classification.

Output layer contains the label which is in the form of one-hot encoded. Now you have a good understanding of CNN.



**Fig 3.10 : Accuracy and Loss Graph**

## MODULE-2(ALEXNET) :

AlexNet, which employed an 8-layer CNN, won the ImageNet Large Scale Visual Recognition Challenge 2012 by a phenomenally large margin. This network showed, for the first time, that the features obtained by learning can transcend manually- designed features, breaking the previous paradigm in computer vision. The architectures of AlexNet and LeNet are very similar the fig shows illustrates. Note that we provide a slightly streamlined version of AlexNet removing some of the design quirks that were needed in 2012 to make the model fit on two small GPUs.

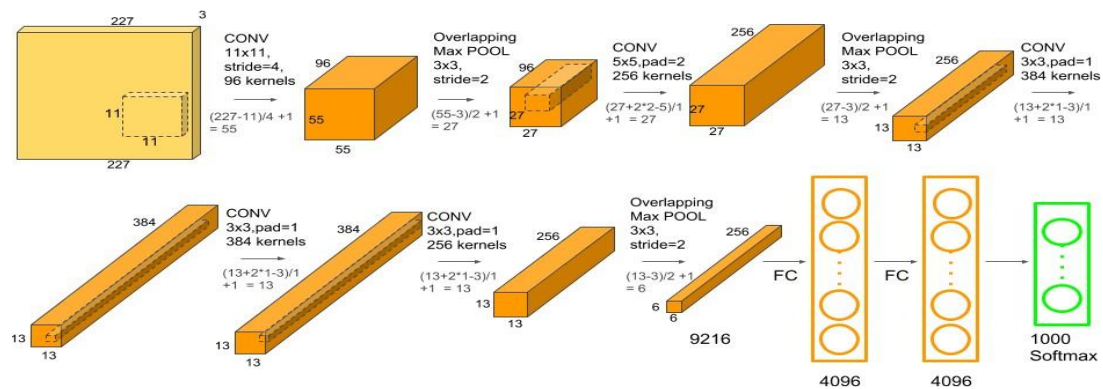


Fig 3.11 : Alexnet Architecture

**Convolutional layer:** A convolution is a mathematical term that describes a dot product multiplication between two sets of elements. Within deep learning the convolution operation acts on the filters/kernels and image data array within the convolutional layer. Therefore a convolutional layer is simply a layer that houses the convolution operation that occurs between the filters and the images passed through a convolutional neural network.

**Batch Normalisation layer:** Batch Normalization is a technique that mitigates the effect of unstable gradients within a neural network through the introduction of an additional layer that performs operations on the inputs from the previous layer. The operations standardize and normalize the input values, after that the input values are transformed through scaling and shifting operations.

**MaxPooling layer** : Max pooling is a variant of sub-sampling where the maximum pixel value of pixels that fall within the receptive field of a unit within a sub-sampling

layer is taken as the output. The max-pooling operation below has a window of 2x2 and slides across the input data, outputting an average of the pixels within the receptive field of the kernel.

**Flatten layer:** Takes an input shape and flattens the input image data into a one-dimensional array.

**Dense Layer:** A dense layer has an embedded number of arbitrary units/neurons within. Each neuron is a perceptron.

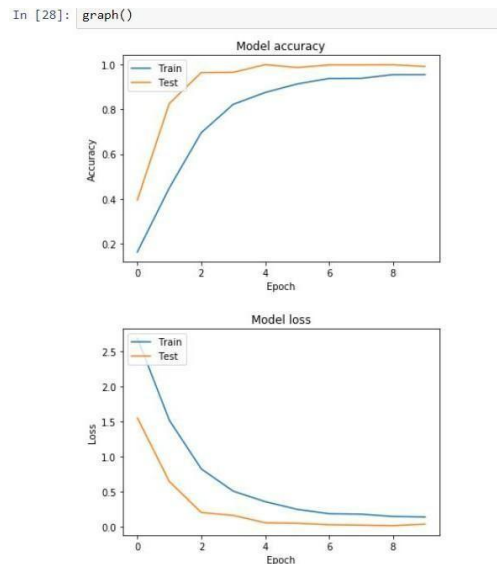
**Some other operations and techniques utilized within the AlexNet CNN that are worth mentioning are:**

**Activation Function :** A mathematical operation that transforms the result or signals of neurons into a normalized output. The purpose of an activation function as a component of a neural network is to introduce non-linearity within the network. The inclusion of an activation function enables the neural network to have greater representational power and solve complex functions.

**Rectified Linear Unit Activation Function(ReLU):** A type of activation function that transforms the value results of a neuron. The transformation imposed by ReLU on values from a neuron is represented by the formula  $y=\max(0,x)$ . The ReLU activation function clamps down any negative values from the neuron to 0, and positive values remain unchanged. The result of this mathematical transformation is utilized as the output of the current layer and used as input to a consecutive layer within a neural network.

**Softmax Activation Function:** A type of activation function that is utilized to derive the probability distribution of a set of numbers within an input vector. The output of a softmax activation function is a vector in which its set of values represents the probability of an occurrence of a class or event. The values within the vector all add up to 1.

**Dropout:** Dropout technique works by randomly reducing the number of interconnecting neurons within a neural network. At every training step, each neuron has a chance of being left out, or rather, dropped out of the collated contributions from connected neurons.



**Fig 3.12 : Accuracy and Loss Graph**

### **MODULE-3 (LENET) :**

In 1998, the LeNet-5 architecture was introduced in a research paper titled “Gradient-Based Learning Applied to Document Recognition” by Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. It is one of the earliest and most basic CNN architecture.

It consists of 7 layers.

The first layer consists of an input image with dimensions of  $32 \times 32$ . It is convolved with 6 filters of size  $5 \times 5$  resulting in dimension of  $28 \times 28 \times 6$ .

The second layer is a Pooling operation which filter size  $2 \times 2$  and stride of 2. Hence the resulting image dimension will be  $14 \times 14 \times 6$ .

Similarly, the third layer also involves in a convolution operation with 16 filters of size  $5 \times 5$  followed by a fourth pooling layer with similar filter size of  $2 \times 2$  and stride of 2. Thus, the resulting image dimension will be reduced to  $5 \times 5 \times 16$ .

Once the image dimension is reduced, the fifth layer is a fully connected convolutional layer with 120 filters each of size  $5 \times 5$ . In this layer, each of the 120 units in this layer will be connected to the 400 ( $5 \times 5 \times 16$ ) units from the previous layers.

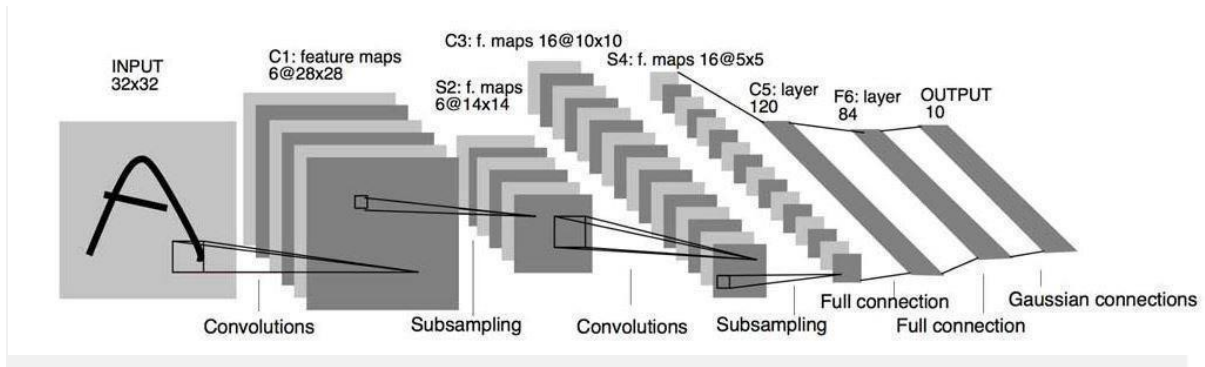
The sixth layer is also a fully connected layer with 84 units.

The final seventh layer will be a softmax output layer with ‘n’ possible classes

depending upon the number of classes in the dataset.

The above diagram is a representation of the 7 layers of the LeNet-5 CNN Architecture.

**LeNet-5**, a 7 layer Convolutional Neural Network, was deployed in many banking systems to recognize hand-written numbers on cheques.



**Fig 3.13 : LeNet Architecture**

Below are the snapshots of the Python code to build a LeNet-5 CNN architecture using keras library with TensorFlow framework

```
#LeNet-5 CNN Architecture
model = keras.Sequential()
model.add(layers.Conv2D(filters=6, kernel_size=(5, 5), activation='relu', input_shape=(32,32,1)))
model.add(layers.AveragePooling2D())
model.add(layers.Conv2D(filters=16, kernel_size=(5, 5), activation='relu'))
model.add(layers.AveragePooling2D())
model.add(layers.Flatten())
model.add(layers.Dense(units=120, activation='relu'))
model.add(layers.Dense(units=84, activation='relu'))
model.add(layers.Dense(units=10, activation = 'softmax'))
```

**Fig 3.14 : LeNet Layers**

In Python Programming, the model type that is most commonly used is the Sequential type. It is the easiest way to build a CNN model in keras. It permits us to build a model layer by layer. The 'add()' function is used to add layers to the model. As explained above, for the LeNet-5 architecture, there are two Convolution and Pooling pairs followed by a Flatten layer which is usually used as a connection between Convolution and the Dense layers.

The Dense layers are the ones that are mostly used for the output layers. The activation used is the 'Softmax' which gives a probability for each class and they

sum up totally to 1. The model will make its prediction based on the class with highest probability.

## DJANGO FRAMEWORK

In this module the trained deep learning model is converted into hierarchical data format file (.h5 file) which is then deployed in our django framework for providing better user interface and predicting the output whether the given fruit which type.

Django is a Python-based web framework that allows you to quickly create efficient web applications. It is also called batteries included framework because Django provides built-in features for everything including Django Admin Interface, default database – SQLite3, etc. When you're building a website, you always need a similar set of components: a way to handle user authentication (signing up, signing in, signing out), a management panel for your website, forms, a way to upload files, etc. Django gives you ready-made components to use and that too for rapid development. Excellent documentation and high scalability. Used by Top MNCs and Companies, such as Instagram, Disqus, Spotify, Youtube, Bitbucket, Dropbox, etc. and the list is never-ending. Easiest Framework to learn, rapid development and Batteries fully included. The last but not least reason to learn Django is Python, Python has huge library and features such as Web Scrapping, Machine Learning, Image Processing, Scientific Computing, etc. One can integrate it all this with web application and do lots and lots of advance stuff.

Django is an MVT web framework that is used to build web applications. The huge Django web-framework comes with so many "batteries included" that developers often get amazed as to how everything manages to work together. The principle behind adding so many batteries is to have common web functionalities in the framework itself instead of adding latter as a separate library.

One of the main reasons behind the popularity of Django framework is the huge Django community. The community is so huge that a separate website was devoted to it where developers from all corners developed third-party packages including authentication, authorization, full-fledged Django powered CMS systems, e-commerce add-ons and so on. There is a high probability that what you are trying to develop is already developed by somebody and you just need to pull that into your project.



**Fig 3.15 : Django Framework**

In a traditional data-driven website, a web application waits for HTTP requests from the web browser (or other client). When a request is received the application works out what is needed based on the URL and possibly information in POST data or GET data. Depending on what is required it may then read or write information from a database or perform other tasks required to satisfy the request. The application will then return a response to the web browser, often dynamically creating an HTML page for the browser to display by inserting the retrieved data into placeholders in an HTML template. Django web applications typically group the code that handles each of these steps into separate files.

- **URLs:** While it is possible to process requests from every single URL via a single function, it is much more maintainable to write a separate view function to handle each resource. A URL mapper is used to redirect HTTP requests to the appropriate view based on the request URL. The URL mapper can also match particular patterns of strings or digits that appear in a URL and pass these to a view function as data.
- **View:** A view is a request handler function, which receives HTTP requests and returns HTTP responses. Views access the data needed to satisfy requests via *models*, and delegate the formatting of the response

to *templates*.

- **Models:** Models are Python objects that define the structure of an application's data, and provide mechanisms to manage (add, modify, delete) and query records in the database.
- **Templates:** A template is a text file defining the structure or layout of a file (such as an HTML page), with placeholders used to represent actual content. A *view* can dynamically create an HTML page using an HTML template, populating it with data from a *model*. A template can be used to define the structure of any type of file; it doesn't have to be HTML!
- Django is a Python-based web framework that allows you to quickly create efficient web applications. It is also called batteries included framework because Django provides built-in features for everything including Django Admin Interface, default database – SQLite3, etc. When you're building a website, you always need a similar set of components: a way to handle user authentication (signing up, signing in, signing out), a management panel for your website, forms, a way to upload files, etc. Django gives you ready-made components to use and that too for rapid development.

Advantages:

Django is a rapid web development framework that can be used to develop fully fleshed web applications in a short period of time.

- It's very easy to switch database in Django framework.
- It has built-in admin interface which makes easy to work with it.
- Django is fully functional framework that requires nothing else.
- It has thousands of additional packages available
- It is very scalable

## **REQUIREMENT SPECIFICATION:**

This proposed software runs effectively on a computing system that has the minimum requirements. Undertaking all the equipment necessities are not satisfied but rather exist in their systems administration between the customer's machines already. So, the main need is to introduce appropriate equipment for the product. The requirements are split into two categories, namely:

### **Software Requirements:-**

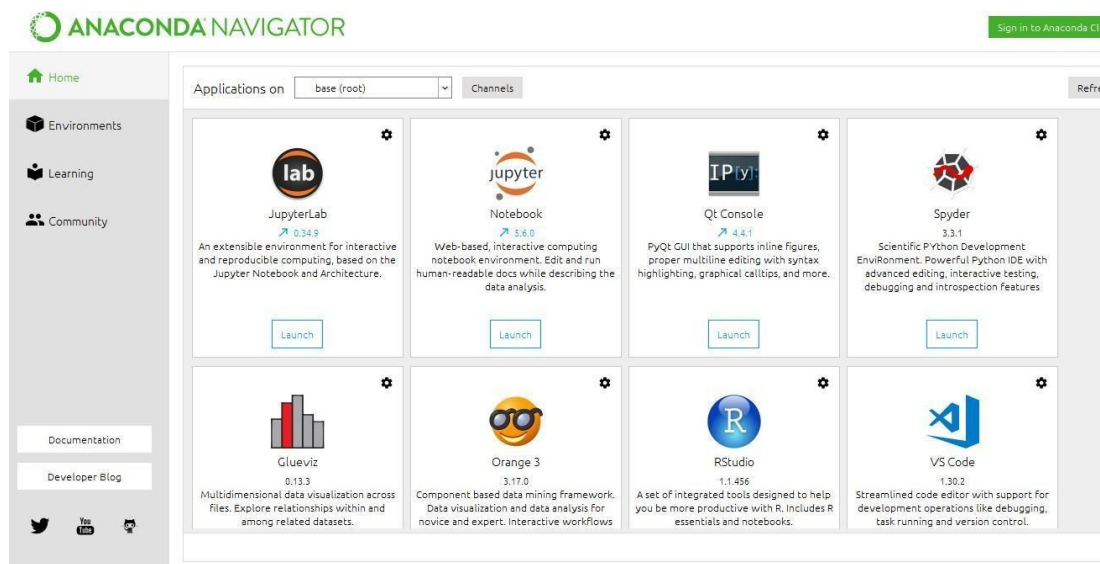
Operating System : Windows/Linux.  
Simulation Tools : Anaconda with Jupyter Notebook, Pycharm.  
Browser : Mozilla Firefox, Safari, chrome, etc.

### **Hardware Requirements:-**

Hard disk : 80 GB and above.  
Processor : Pentium IV or III.  
Ram : Minimum 2GB.

## CHAPTER 4

### RESULT AND DISCUSSION



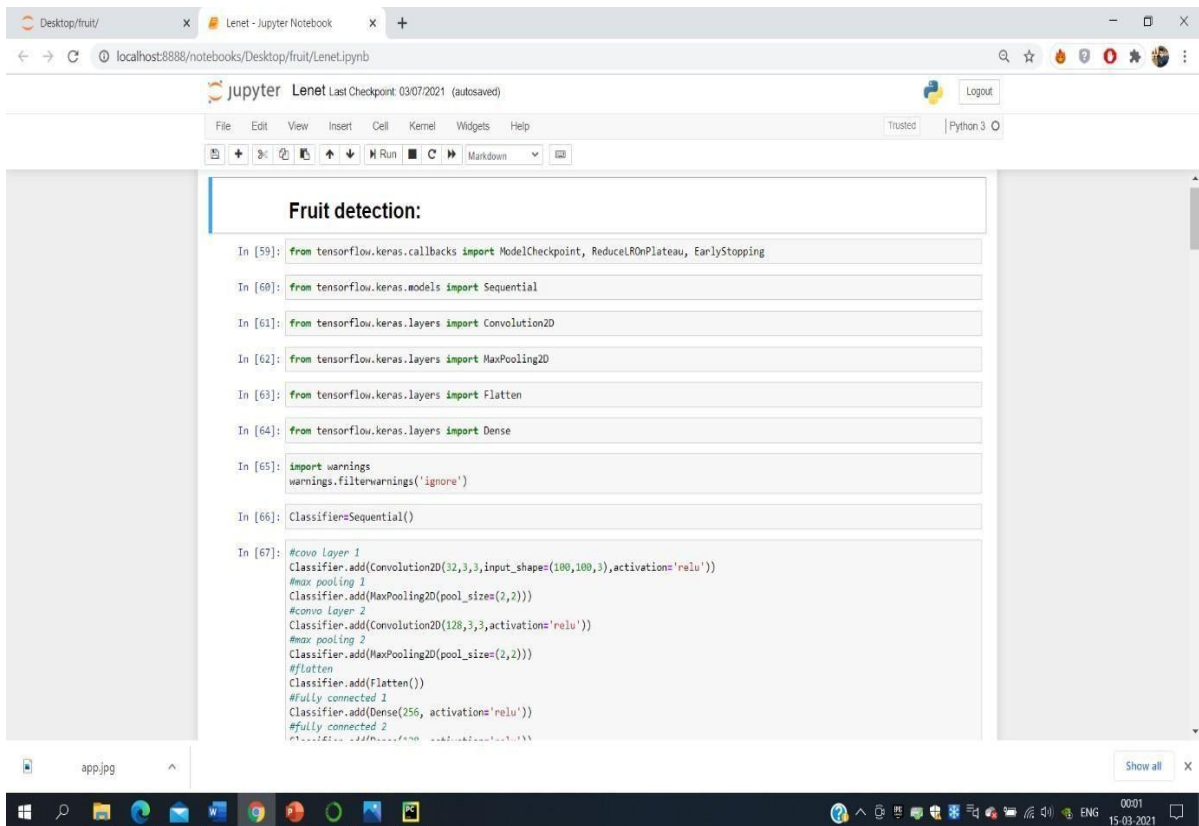
**Fig 4.1 : Anaconda navigator**

The Fig 4.1 above shows Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository. It is available for Windows, macOS, and Linux. Open this navigator and launch the Jupyter Notebook to train the CNN models and the model Accuracy is shown in graphs.

The Jupyter Notebook which is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

This platform all graphical user interface which has application to easily manage conda packages. The next is the Lenet CNN model is trained in the jupyter notebook. And the training and fitting the model is done here and it has high

accuracy of detecting the fruits and vitamins present in it. The model is trained in such a way to detect the fruits in offline mode.

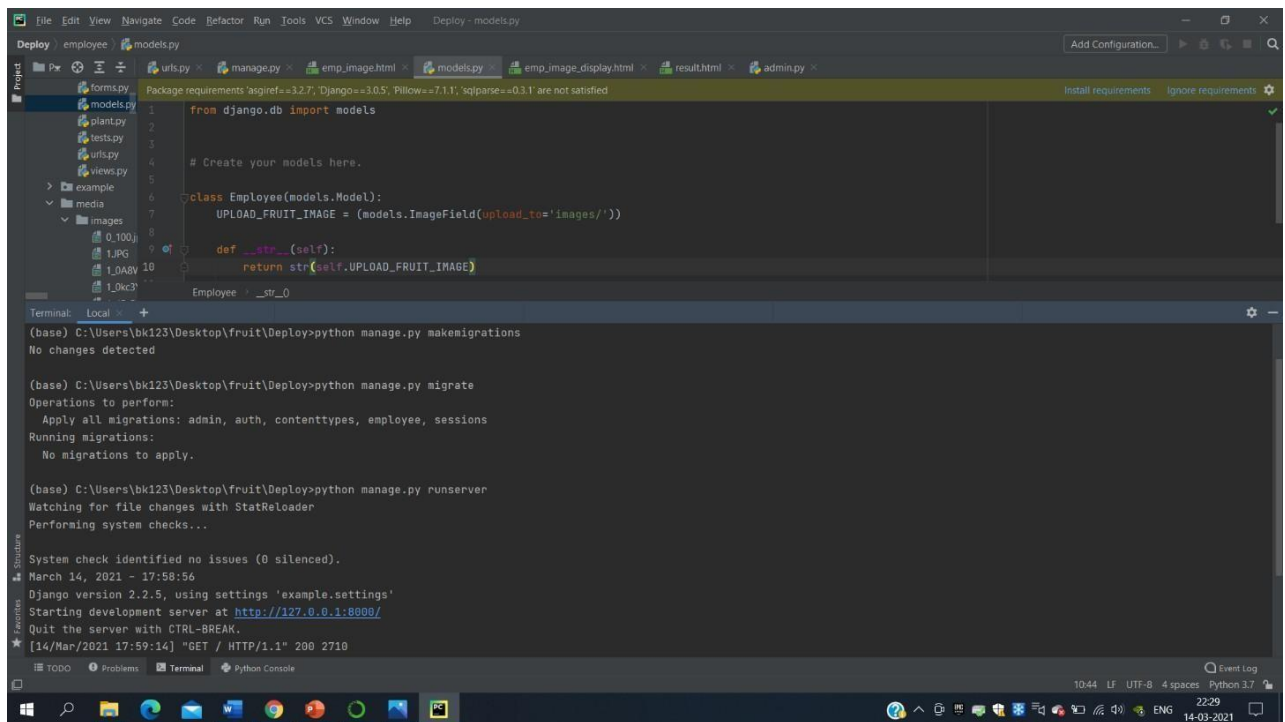


**Fig 4.2 : LeNet model**

This Fig 4.2 shows the Jupyter Notebook platform in which we have created the model and trained the dataset. The LeNet Convolutional Neural Network has highest accuracy compared to manual CNN and AlexNet CNN. The classifier is saved in Heirarichal format and the model is deployed in Pycharm Software in which the web application is created for better user interface.

The Lenet model is consists of 2 covolutional layer and 2 pooling layer and 2 fully connected layer followed by 1 output layer with softmax activation. The relu activation function is used to remove the negative values and replace them with 0. Dense is the layer where the input is passed by this layer to output layer for detection.

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda. PyCharm is cross platform, with Windows, macOS and Linux versions.



**Fig 4.3 : Pycharm Screen**

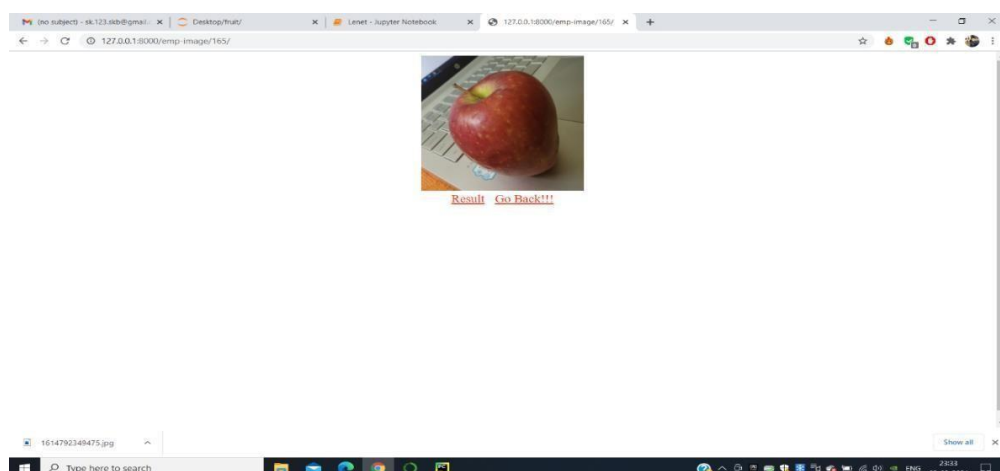
As we have already implemented the Lenet Convolutional Neural Network model in Django framework, we use Pycharm to run the model. Here the deployment of the model is loaded here and in the cammand terminal we need type the cammand “python manage.py makemigrations” after no changes detected we need to type another command python manage.py migrate it will apply to all migrations. Then at last we need to type the command which is “python manage.py runserver” in the terminal window. This command you'll probably run the most of all commands. It means to run a emulated server on your local computer. So, after running it, you can go to (http://localhost:8000) or [127.0.0.1:8000](http://127.0.0.1:8000).



**Fig 4.4 : Output as web application**

After entering into the link in Pycharm, the local host page will open. Here local host is created using Django framework. The HTML and CSS is used as frontend and Sqlite3 as backend. This Home page is displayed as "FRUITS DETECTION USING ARTIFICIAL INTELLIGENCE". The webpage is designed by background image using gifs or jpg format.

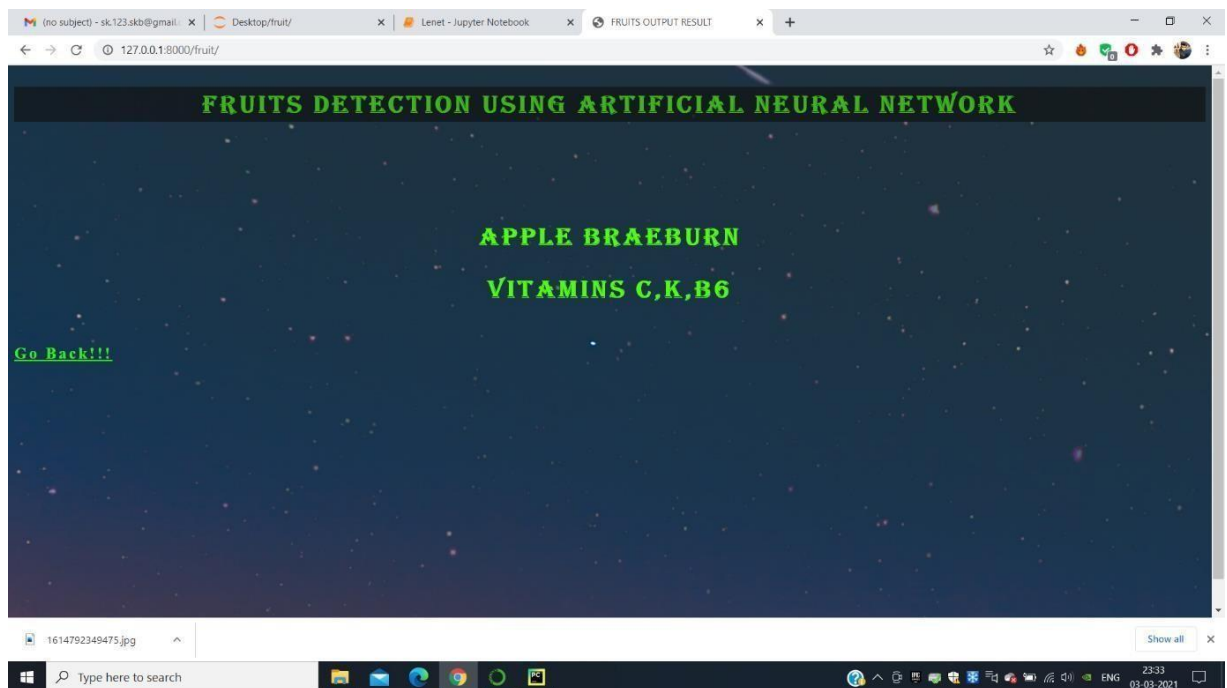
Inside the home screen it displays "UPLOAD IMAGE HERE" followed by file location to upload the image which is present in system.. And that field is mandatory for detecting the fruit. And finally UPLOAD button is triggered the selected image will be shown in next page as intermediate screen.



**Fig 4.5: Displaying Image Screen**

Next is the intermediate screen in which the uploaded image is seen. The uploaded image is a real-time photograph that has been shot on mobile is displayed here in Fig 4.5 as a preview and has 2 buttons "result" and "Go back" button.

The result is used as a button to see the result of the image and the user wants to check another fruit they can use "Go back" button to display next fruits name when we click the result button it directly tells the fruit type and vitamins present in next page.



**Fig 4.6 : Final Output**

The Fig 4.6 shows the Final Output of the uploaded image tells the fruit name and vitamins present in it. The title will be displayed as "FRUITS DETECTION USING ARTIFICIAL NEURAL NETWORK" with the blink style.

The image which we want to detect is seen as the text format of fruits name and vitamins present is seen here as the final output. The "Go Back" button is used for next fruits name to detect or predict.

## **CHAPTER 5**

### **CONCLUSION AND FUTURE WORK**

The proposed project is able to detect the fruits type and vitamins. This helps us to gain some extra knowledge. Fruit detection will be able to reduce the current ongoing problems. It reduces confusion among the particular fruit. Its an offline application. To identify the fruits type based on artificial neural network. It is the best deep learning model to identify the fruits types and it shows how much of vitamins present. We described a new and complex database of images with fruits. Also we made some numerical experiments by using TensorFlow library in order to classify the images according to their content. From our point of view one of the main objectives for the future is to improve the accuracy of the neural network.

This involves further experimenting with the structure of the network. Various tweaks and changes to any layers as well as the introduction of new layers can provide completely different results. Another option is to replace all layers with convolutional layers. This project has been shown to provide some improvement over the networks that have fully connected layers in their structure. A consequence of replacing all layers with convolutional ones is that there will be an increase in the number of parameters for the network [4]. Another possibility is to replace the rectified linear units with exponential linear units.

This project is mainly concentrating in reducing human effort, and to provide a better interface. The vitamins in the fruits are very important to stay healthy. Some people may search fruits which has specific vitamins to gain energy. Sorting the fruits may be difficult and time consuming by using this method we can reduce time and labour work. We would like to try out these practices and also to try to find new configurations that provide interesting results.

Tasks to be done in future work:

- In the near future we plan to create a mobile application which will take pictures of fruits and labels them accordingly. To deploy this model as mobile application.
- To implement this model to an AI using voice command feedback.

- To implement this model to automated machinery for fruit sorting.

## REFERENCES

- [1] Aifeng Ren, Adnan Zahid, Ahmed Zoha, Syed Aziz Shah, Muhammad Ali Imran, Akram Alomainy, Qammer H.Abbasi " Machine Learning-Driven Approach Towards the Quality Assessment of Fresh Fruits Using Non-invasive Sensing" IEEE Sensors Journal, vol. 20, no. 4,pp.2075-2083, Feb.15, 2020.
- [2] Enrico Bellocchio, Thomas A. Ciafuglia, Gabriele Costante, Paolo Valigi" Weakly Supervised Fruit Counting for Yield Estimation Using Spatial Consistency" IEEE Robotics and Automation Letters, vol. 4, no.3, pp. 2348-2355, March 06, 2019.
- [3] Gianluigi Ciocca,Giovanni Micali,Paolo Napoletano" State Recognition of Food Images Using Deep Features" IEEE Access,vol. 8,pp. 32003-32017,Feb 2020.
- [4] Guantao Xuan, Chong Gao, Yuanyuan Shao, Meng Zhang, Yongxian Wang, Jingrun Zhong, Qingguo Li"Apple Detection in Natural Environment Using Deep Learning Algorithms" IEEE Access, vol 8,pp-216772 – 216780, Nov 2020.
- [5] Guoxiang Zeng "Fruit and vegetables classification system using image saliency and convolutional neural network" 2017 IEEE 3rd Information Technology and Mechatronics Engineering, Dec 2017.
- [6] Muladi Muladi, Dyah Lestari, Dedy Tri Prasetyo" Classification Of Eligibility Consumption Of Manalagi Apple Fruit Varieties Using Backpropagation" 2019 International Conference on Advanced Mechatronics, Intelligent Manufacture and Industrial Automation (ICAMIMIA), Oct 2020.
- [7] Shiyu Luo, Kamal Sarabandi, Ling Tong, Leland Pierce"A SAR Image Classification Algorithm Based on Multi-Feature Polarimetric Parameters Using FOA and LS-SVM" IEEE Access, vol. 7,pp. 175259-175276,Dec 04,2019.
- [8] Wilson Castro, Jimy Oblitas, Miguel De-La Torre, Carlos Cotrina, Karen Bazan, Himer Avila George "Classification of Cape Gooseberry Fruit According to its Level of Ripeness Using Machine Learning Techniques and Different Color

Spaces" IEEE Access, vol. 7,pp. 27389-27400,Mar 04,2019.

- [9] Yingqiong Peng, Muxin Liao, Yuxia Song, Zhichao Liu, Huojiao He, Hong Deng, Yinglong Wang "FB-CNN: Feature Fusion-Based Bilinear CNN for Classification of Fruit Fly Image IEEE Access, vol. 8, pp. 2169-3536,Dec 2019.
- [10] Yo-Ping Huang, Tzu-Hao Wang, Haobijam Basanta "Using Fuzzy Mask R-CNN Model to Automatically Identify Tomato Ripeness" IEEE Access, vol. 8,pp. 207672- 207682,Nov 2020.

## APPENDIX

### A. SAMPLE CODE:

```
import os
import numpy as np # linear algebra
import matplotlib.pyplot as plt
# DL framwork - tensorflow, keras a backend
import tensorflow as tf
import tensorflow.keras.backend as K
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.layers import Input, Dense, Flatten, Dropout,
BatchNormalization
from tensorflow.keras.layers import Conv2D, SeparableConv2D, MaxPool2D,
LeakyReLU, Activation
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau,
EarlyStopping
from IPython.display import display
from os import listdir
from os.path import isfile, join
from PIL import Image
import glob

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Convolution2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
Classifier.add(Convolution2D(32,3,3,input_shape=(100,100,3),activation='relu'))
Classifier.add(MaxPooling2D(pool_size=(2,2)))
Classifier.add(Convolution2D(128,3,3,activation='relu'))
Classifier.add(MaxPooling2D(pool_size=(2,2)))
```

```
Classifier.add(Flatten())
```

```
Classifier.add(Dense(256, activation='relu'))
```

```

Classifier.add(Dense(27, activation='softmax'))

Classifier.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['ac
curacy'])
Classifier.summary()
Classifier.add(Dense(29, activation='softmax'))
Classifier.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['ac
curacy'])
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_rang
e=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
training_set=train_datagen.flow_from_directory('fruits-
360/train',target_size=(100,100),batch_size=32,class_mode='categorical')
test_set=test_datagen.flow_from_directory('fruits-
360/test',target_size=(100,100),batch_size=32,class_mode='categorical')
img_dims = 150
epochs = 4
batch_size = 32
#### Fitting the model
history = Classifier.fit_generator(
    training_set, steps_per_epoch=training_set.samples // batch_size,
    epochs=epochs,
    validation_data=test_set,validation_steps=test_set.samples // batch_size)
def graph():
    #Plot training & validation accuracy values
    plt.plot(history.history['accuracy'])
    plt.plot(history.history['val_accuracy'])
    plt.title('Model accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend(['Train', 'Test'], loc='upper left')

```

```
plt.show()
```

```
# Plot training & validation loss values
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()
```

```
import os
import numpy as np # linear algebra
import matplotlib.pyplot as plt
# DL framework - tensorflow, keras a backend
import tensorflow as tf
import tensorflow.keras.backend as K
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.layers import Input, Dense, Flatten, Dropout,
BatchNormalization
from tensorflow.keras.layers import Conv2D, SeparableConv2D, MaxPool2D,
LeakyReLU, Activation
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau,
EarlyStopping
from IPython.display import display
from os import listdir
from os.path import isfile, join
from PIL import Image
import glob
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Convolution2D
```

```
from tensorflow.keras.layers import MaxPooling2D  
from tensorflow.keras.layers import Flatten
```

```

from tensorflow.keras.layers import Dense
model = Sequential()
# 1st Convolutional Layer
model.add(Conv2D(filters=96, input_shape=(100,100,3), kernel_size=(11,11),
strides=(4,4), padding='valid'))
model.add(Activation('relu'))
) # Max Pooling
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2), padding='valid'))

model.add(Flatten())
# 1st Fully Connected Layer
model.add(Dense(500, input_shape=(224*224*3,)))
model.add(Activation('relu'))
# Add Dropout to prevent overfitting
model.add(Dropout(0.4))
# 2nd Fully Connected Layer
model.add(Dense(250))
model.add(Activation('relu'))
# Add Dropout
model.add(Dropout(0.4))
# 3rd Fully Connected Layer
model.add(Dense(112))
model.add(Activation('relu'))
# Add Dropout
model.add(Dropout(0.4))
# Output Layer
model.add(Dense(29))
model.add(Activation('softmax'))
model.summary()
# Compile the model
model.compile(loss = 'categorical_crossentropy', optimizer='adam',

```

```
metrics=['accuracy'])
```

```
Classifier.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['accuracy'])
Classifier.summary()
Classifier.add(Dense(29, activation='softmax'))
Classifier.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['accuracy'])
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
training_set=train_datagen.flow_from_directory('fruits-360/train',target_size=(100,100),batch_size=32,class_mode='categorical')
test_set=test_datagen.flow_from_directory('fruits-360/test',target_size=(100,100),batch_size=32,class_mode='categorical')
img_dims = 150
epochs = 4
batch_size = 32
#### Fitting the model
history = Classifier.fit_generator(
    training_set, steps_per_epoch=training_set.samples // batch_size,
    epochs=epochs,
    validation_data=test_set,validation_steps=test_set.samples // batch_size)
```