A project report

on

IDENTIFYING THE HEALTH CONDITION OF A LEAF USING CNN

Submitted in partial fulfillment of the

requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

by

B.DRAKSHAYINI	**184G1A0516**
N.KEERTHI	**184G1A0530**
G.NAVEEN KUMAR	**184G1A0552**
K.KARTHIK	**194G5A0503**

Under the Guidance of

Mr. P. Veera Prakash M.Tech, (Ph.D)

Assistant Professor

Department of Computer Science & Engineering

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)

(Accredited by NAAC with 'A' Grade & Accredited by NBA(EEE, ECE & CSE))
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.

2021-2022

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)
(Accredited by NAAC with 'A' Grade & Accredited by NBA (EEE, ECE & CSE))
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Certificate

This is to certify that the Technical Seminar report entitled IDENTIFYING THE HEALTH CONDITION OF A LEAF USING CNN is the bonafide work carried out by **B. Drakshayini,** bearing Roll Number **184G1A0516, N. Keerthi,** bearing Roll Number **184G1A0530, G. Naveen Kumar,** bearing Roll Number **184G1A0552, K. Karthik,** bearing Roll Number **194G5A0503** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2021-2022.

Signature of the Guide	**Head of the Department**
Mr. P. Veera Prakash M. Tech, (Ph. D)	Mr. P. Veera Prakash M. Tech, (Ph. D)
Assistant Professor	Assistant Professor

Date: **EXTERNAL EXAMINER**

Place: Rotarypuram

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we now have the opportunity to express our gratitude for all of them.

It is with immense pleasure that we would like to express our indebted gratitude to our Guide **Mr. P. Veera Prakash** **M. Tech, (Ph. D), Assistant Professor & Head Of the Department, Computer Science & Engineering**, who has guided us a lot and encouraged us in every step of the project work. We thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project.

We express our deep felt gratitude to **Mr. K. Venkatesh** **M.Tech, Assistant Professor,** project coordinator valuable guidance and unstinting encouragement enabled us to accomplish our project successfully in time.

We are very much thankful to **Mr. P. Veera Prakash** **M.Tech, (Ph. D), Assistant Professor & Head Of the Department, Computer Science & Engineering,** for his kind support and for providing necessary facilities to carry out the work.

We wish to convey our special thanks to **Dr. G. Bala Krishna** **M. Tech, Ph. D, Principal** of **Srinivasa Ramanujan Institute of Technology** for giving the required information in doing our project work. Not to forget, we thank all other faculty and non-teaching staff, and our friends who had directly or indirectly helped and supported us in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities**.**

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

Project Associates

DECLARATION

We, B. Drakshayini bearing reg no: 184G1A0516, N. Keerthi bearing reg no: 184G1A0530, G. Naveen Kumar bearing reg no: 184G5A0552, K. Karthik bearing reg no: 194G5A0503, students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram, hereby declare that the dissertation entitled "IDENTIFYING THE HEALTH CONDITION OF A LEAF USING CNN" embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of Mr. P. Veera Prakash M.Tech, (Ph.D), Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities of Institute for the award of Degree.

B. DRAKSHAYINI	Reg no: 184G1A0516
N. KEERTHI	Reg no: 184G1A0530
G. NAVEEN KUMAR	Reg no: 184G1A0552
K. KARTHIK	Reg no: 194G5A0503

CONTENTS **Page No**

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Networks
SVM	Support Vector Machines
ANN	Artificial Neural Networks
FCL	Fully Connected Layers
PCA	Principal Component Analysis
UML	Unified Modeling Language
GUI	Graphical User Interface

ABSTRACT

Early disease detection and pests are important for better yield and quality of crops. With reduction in quality of the agricultural product, diseases in plants can lead to the huge economic losses to the individual farmers. In country like India whose major population is involved in agriculture, it is very important to find the disease at early stages. Faster and precise prediction of plant disease could help reducing the losses. With the significant advancement and developments in deep learning have given the opportunity to improve the performance and accuracy of detection of object and recognition system. Deep learning (sometimes known as "deep structured learning") is a machine learning method that is part of a broader family.

It's possible to learn under close supervision, in a semi-unsupervised setting, or on one's own. It is very difficult to monitor the plant diseases manually. It requires tremendous amount of work, expertise in the plant diseases, and also require the excessive processing time. Hence, Convolutional Neural Networks can be used for the detection of plant health condition. Deep learning techniques, and in particular Convolutional Neural Networks (CNNs), have led to significant progress in image processing. Since 2016, many applications for the automatic identification of crop diseases have been developed. These applications could serve as a basis for the development of expertise assistance or automatic screening tools. To build disease recognition model we are applying RESNET CNN model. It involves the steps like Image Acquisition, Image Pre-Processing, Image Segmentation, Feature Extraction and Classification into healthy or diseased leaves.

Keywords: plant disease, image processing, deep learning, convolutional neural networks

CHAPTER-1
INTRODUCTION

In India and around the world, agricultural production is the primary source of nutrition. Throughout the course of its development, it is plagued by a wide range of ailments. A high-quality crop depends on early diagnosis and treatment of these diseases, but this is challenging due to the enormous tracts of land controlled by individual farmers, the great variety of diseases they carry, and the likelihood of many diseases in a single crop. It's tough and time-consuming to locate agricultural experts in remote areas.

As a result, Automated Systems are required. To help farmers in need and improve the precision with which plant's health condition can be identified, Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs) have been utilized in studies[1,4]. But the accuracy of these systems depends greatly on the features that are selected. Images may now be recognized using Convolutional Neural Networks, which eliminates the requirement for picture preprocessing and provides built-in feature selection. Another issue is that huge datasets for these kinds of challenges are extremely difficult to come by. In situations where the dataset size is relatively small, it is preferable to use a model that has been trained on a large dataset. When developing a new model using Transfer Learning, it is possible to delete the final layer of connections or to fine tune the final layers to be more specific to the dataset under consideration.

Farmers can upload photographs of infected leaves taken using their smartphones in our application , where our neural network will identify the type of plant and its health condition whether it is healthy or not. Convolutional Neural Network work has helped us build a deep learning technique in this work. Transfer Learning has been utilized to fine-tune the fully connected layers so that we can fit our own datasets into the Resnet50 model's fully connected layers. At the end, we looked at our mistakes and tried to figure out why they happened.

1.1 Objectives and Scope of this Project

1.1.1 Objectives

In the Indian economic sector, agriculture plays an important role and India contributes a fourth place in agricultural production. Approximately all the states in India practise cultivation like Tamil Nadu, West Bengal, Punjab, Uttar Pradesh, Assam, Bihar, etc. Agriculture sector is providing its share about 19.9% to the total gross domestic product. The different varieties of diseases may occur in individual crop which is difficult to identify to the farmers with their limited knowledge gained through the experience. For this precision and early identification of plant diseases diagnosis, automatic data processing expert system is crucially requisite. Thus the healthy and successful cultivation is viable.

1.1.2 Scope of the Project

In this study, a pre-trained ResNet50 model incorporates the knowledge in the form of weights which is transformed to our experiment for the feature extraction process by using transfer leaning approach. The deep learning model has been fine-tuned for getting the improved accuracy in identification of the plant’s health condition.

CHAPTER-2
METHODOLOGY

The architecture of Convolutional Neural Networks (CNNs) dictates the network's performance. The convolution layer, the pooling layer, and the fully linked layer are all part of it[2]. Three layers are involved in this process: a feature extractor (the first two) and a classification layer (the third). Convolutional features are reduced in dimension by the pooling layer. The features extracted by the fully connected layer are then used by softmax to classify the images. The learnable filters in the convolution layer retrieve the information from the input image.

The two-dimensional feature map is created by taking the dot product of each filter with the raw picture pixel in a sliding window. The ReLU is one of the most common activation functions. Feature maps can be reduced by using "max pooling", a subsampling layer. It's then possible to connect each feature map to the completely connected layer. Using softmax, each class is assigned a decimal probability.

By using pre-trained models, it is possible to move weights, architecture, and features from one domain to another. There are more data-specific features in the upper layers. The last dense layer is used for classification, followed by a softmax layer. In comparison to prior techniques, CNN is a feature extractor, which gives it an edge over CNN. We use our dataset to retrain the model and fine-tune it in order to get the classifications.

Given the size and familiarity of the new dataset, four transfer learning procedures are possible. Small and familiar, the new dataset is derived from the old. Unlike the previous dataset, the new dataset is both huge and familiar. Even though it's a minor addition, the new data is quite distinct from the old. It's a massive change from the last one, and it's completely new. It follows that in order to employ the ResNet as a feature extractor until the final fully connected layers that were fine-tuned in order to meet the number of classes in our dataset, our model falls into the third situation as outlined above. Details of the proposed architecture are displayed. Each of the four convolution layers is followed by a ReLU, Maxpooling, and dropout layer, and then two Fully Connected Layers (FCL) and Softmax are added on top of that.

CHAPTER-3

LITERATURE SURVEY

3.1 Using Deep Learning for Image-Based Leaf Health Condition Detection

Authors: V. Singh, A. Misra

Abstract: Lack of infrastructure makes it difficult to detect plant's health condition rapidly, despite their considerable threat to global food security. Because of the extensive usage of smartphones around the world and recent improvements in computer vision made possible by deep learning, disease diagnosis by smartphone is becoming a reality. We train a convolutional neural network to recognise 14 crop species and 26 illnesses from a publically available dataset using 54,306 pictures of damaged and healthy plant leaves (or absence thereof)[3].

Using a held-out test set, the trained model achieves an accuracy of 99.35%, demonstrating the validity of the approach. Using a set of shots that were not taken in the same conditions as those used for training, the model's accuracy is just 31.4 percent The total accuracy can still be improved by using a more diversified set of training data, though. For crop disease diagnosis on a global scale, training deep learning models on increasingly large and publically available image datasets provides a clear path.

3.2 SVM Classifier Based Grape Leaf Health Condition Detection

Authors: P. B. Padol and A. A. Yadav

Abstract: Grapes are one of India's most popular fruit crops. Grape yields fall as a result of disease infections on the fruit, stems, and leaves. Toxins from bacteria, fungus or virus are the most common causes. There are many factors that limit the amount of fruit that can be produced, including illness. It's impossible to implement effective control measures without an accurate diagnosis of the disease. For the identification of health condition of a plant, image processing is a commonly employed technique. SVM classification is used in this work to aid in the detection and categorization of grape leaf health condition. Segmentation by K-means clustering is used to locate the sick area, and then colour and texture features are extracted. Leaf health condition can finally be identified using a classification system. Accuracy in the suggested approach is 88.89 percent for the condition being studied[4].

3.3 Very Deep Convolutional Networks for Large Scale Image Recognition

Authors: Karen Simonyan, Andrew Zisserman

Abstract: Using a large-scale image identification task, we investigate how a convolutional network's depth influences its performance. Using an architecture with extremely small (3x3) convolution filters, we examined networks of increasing depth and found that raising the depth to 16-19 weight layers significantly improved performance over prior-art setups. This is the main contribution of our study[5]. As a result of these discoveries, our team was able to take first and second place in the 2014 ImageNet Challenge in the categories of localization and classification. Two of our finest ConvNet models have been provided so that further research into deep visual representations in computer vision can take place.

3.4 Hyper-Class Augmented and Regularized Deep Learning for Fine-Grained Image Classification

Authors:S. Xie, T. Yang, X. Wang, and Y. Lin

Abstract: Deep convolutional neural networks have shown significant success in large-scale object recognition (CNNs). Due to the high cost of fine-grained labelled data (usually necessitating domain expertise), as well as the huge intra-class and moderate inter-class variance, fine-grained image classification (FGIC) is significantly more challenging than generic object identification. Using an external dataset (like ImageNet) to pre-train the CNN and then fine-tuning it on the small target data to meet a specific classification goal is one of the most prevalent ways.

Two new aspects of the problem of learning a deep CNN are introduced in this paper: identifying easily annotated hyper-classes inherent in the fine-grained data and collecting numerous images with hyper-classes labelled from readily available external sources (such as image search engines), formulating the problem as multitask learning; and (iii) a novel learning modality, which is based on the idea of a "multitask learning" paradigm. Two small, fine-grained datasets (Stanford Dogs and Stanford Cars) as well as a big, automotive-specific dataset have all been used to evaluate the proposed approach[6].

CHAPTER-4
SYSTEM ANALYSIS

4.1 Existing System

In the past, the only way to identify a disease was to manually examine the leaf. The illness was discovered using a combination of visual inspection of plant leaves and consulting a reference book. This method has three key drawbacks: limited accuracy, the inability to analyze every leaf, and a lengthy time investment. As science and technology progress, new methods for accurately diagnosing plant’s illness emerge. Image processing and deep learning are two methods to consider. Filtering, clustering and histogram analysis are some of the approaches used in image processing to identify the diseased area. Deep learning neural networks, on the other hand, are used to detect disorders.

4.1.1 Disadvantages of Existing System

- It's impossible to study every leaf, and it takes a long time.
- Manual detection requires high domain knowledge.
- It is expensive.

4.2 Proposed System

A ResNet50 transfer learning neural network is used in this project to train a dataset of leaf images, and the trained model may be used to predict health condition from new photos. We used the transfer learning CNN algorithm, which transfers an existing CNN model to a new dataset and then uses the new data to train the model. It has been shown that Resnet50 transfer learning improves prediction accuracy in both a normal CNN model and a normal CNN model with ResNet50 transfer learning.

4.2.1 Advantages of Proposed System

ResNet50 can be utilized to identify the plant's health condition. Over ninety-five percent of the time, it has been right.

4.3 System Requirements

4.3.1 Software Requirements

Product viewpoint and features, OS and operating environment, graphics requirements, design limitations and user documentation are all included in the functional requirements or overall description documents.

- Python IDE 3.7 version (or)
- Anaconda 3.7 (or)
- Jupyter Notebook

4.3.2 Hardware Requirements

Enthought Python / Canopy / VS Code users hardware requirements are highly dependent on the software they are developing. Large arrays/objects in memory demand more RAM, but apps that need to do several calculations or jobs quickly require a faster processor.

• Operating System	: Windows, Linux
• Processor	: Minimum Intel i3
• Ram	: Minimum 4GB
• Hard Disk	: Minimum 250GB

4.4 Functional Requirements

- Data Collection
- Data Pre-Processing
- Training and Testing

- Modelling
- Predicting

4.5 Non-Functional Requirements

A software system's quality trait is known as Non-Functional Requirement (NFR). These non-functional criteria are crucial to the success of the software system and are judged on their responsiveness, usability, security, and portability. “How quickly does the website load?” is an example of a nonfunctional demand. Systems that fall short of user expectations may be the result of a failure to meet non-functional requirements. Through the use of non-functional requirements, it is possible to place limitations on the system’s design in various agile backlogs. With a concurrent user count of more than 10,000, the site should be able to load in three seconds. Functional and non-functional criteria are equally important[7].

- A requirement for ease of use; a requirement for ease of maintenance;
- The need for manageability
- In order to meet the recovery criteria,
- The need for security
- To meet the Data Integrity standard,
- Demand for capacity
- Requirement of availability
- Need for scalability
- The need for interoperability
- Reliability is essential.
- The demand for upkeep
- Legislative mandate
- In terms of the environment, this is an important consideration.

4.6 System Study

Feasibility Study

Project feasibility and cost estimates are calculated at this point, and a business proposal is drafted to go along with it. The proposed system’s feasibility will be studied during system analysis. As a precautionary measure, there will be no additional costs to the company as a result of the proposed system. Feasibility studies necessitate a grasp of the system's fundamental requirements. Three key components make up the feasibility analysis.

Flexibility of the Market

In terms of technical feasibility, this is the most important factor.

Flexibility in Social Context

An evaluation of the system's financial impact on the organization is the purpose of this research.

Flexibility in Technical Terms

The purpose of this investigation is to see if the system's technical criteria can be met. A system's technological resources shouldn't be taxed beyond their capacity when it is under development. As a result, the limited technical resources will be put under a lot of stress. This will put a lot of pressure on the customer. The designed system must have a low demand because only little or no adjustments are needed to implement this system.

Social Comfort

The study's main objective is to determine how well the system is perceived by the intended audience it is designed for. Educating the user on how to get the most out of the technology falls under this category. The system should not be viewed as a threat, but rather as a need. The methods employed to educate and acclimate the user to the system have the exclusive influence on the level of user acceptance. In order for him to be able to provide some useful input as the system's ultimate consumer, his self-esteem must be boosted.

Related Work

Many studies have used classical classifiers, however the outcomes are dependent on the feature selection approaches and picture preprocessing is a significant stage in the process of study. There are numerous studies that are taking use of CNN's high recognition accuracy because of that.

Detection of Plant Health Condition using CNN Paper-based training uses 20,638 photos from 3 plant species. One of the best models has a 99.53% success rate in correctly classifying objects. When evaluated on a different dataset derived from a real-world event, the 99.35 percent success rate dropped to 31.4%. Determining the severity of a condition is more difficult than just classifying it. Because of the high levels of intra-class resemblance across photos belonging to the same class, it is even more difficult to identify it.

AlexNet is used to train the classifier. 91.23% of the time, the aforementioned architecture is able to accurately forecast whether or not a leaf is unhealthy. Le-Net and AlexNet were used as inspiration for the design of its architecture, which was tested and achieved 95.48% accuracy. Pre-processing steps including image resizing to 512*512, normalisation, PCA and whitening were performed because the data was so sparse. Instead of using max pooling, they opted for stochastic pooling, which they claimed prevented over fitting.

CHAPTER-5

SYSTEM DESIGN

5.1 System Architecture

For convolutional neural networks, the ResNet50 CNN design is used. It remains one of the greatest vision model designs to this day. Deep residual networks like the popular ResNet-50 model is a convolutional neural network (CNN) that is 50 layers deep. A Residual Neural Network (ResNet) is an Artificial Neural Network of a kind that stacks residual blocks on top of each other to form a network.

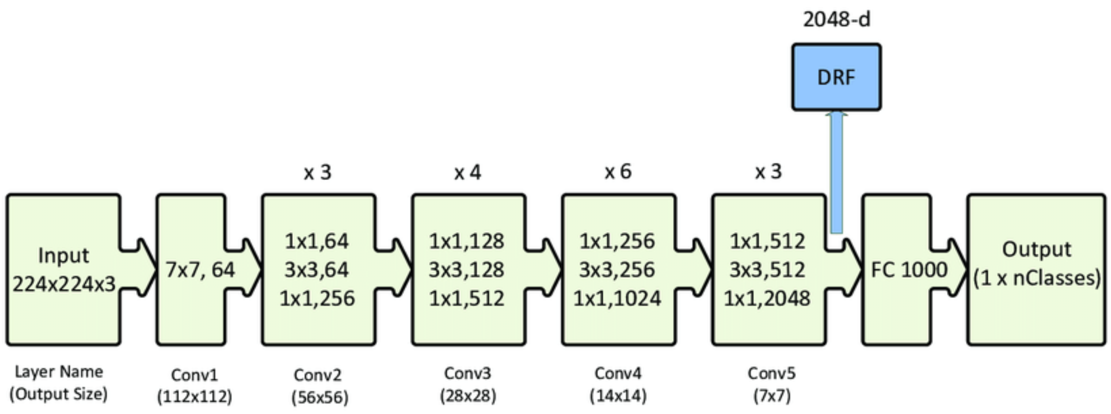


Fig 5.1. Architecture of ResNet50

5.2 Data Flow Diagram

1. The DFD can also be referred to as bubble charts. For example, a simple visual formalism can portray a system in terms of the input and output of data.

2. Second, the data flow diagram is a crucial modeling tool (DFD). Models the system’s components. System processes, data, an external entity interacting with the system, and information flow within the system are all included in this list.

3. The DFD displays how information flows through the system and how it is transformed through a sequence of transformations. It’s a visual representation of how data goes from input to output and the transformations that take place along the way.

4. DFD, or bubble chart, is a type of graph. A DFD can be used at any degree of abstraction to represent a given system.

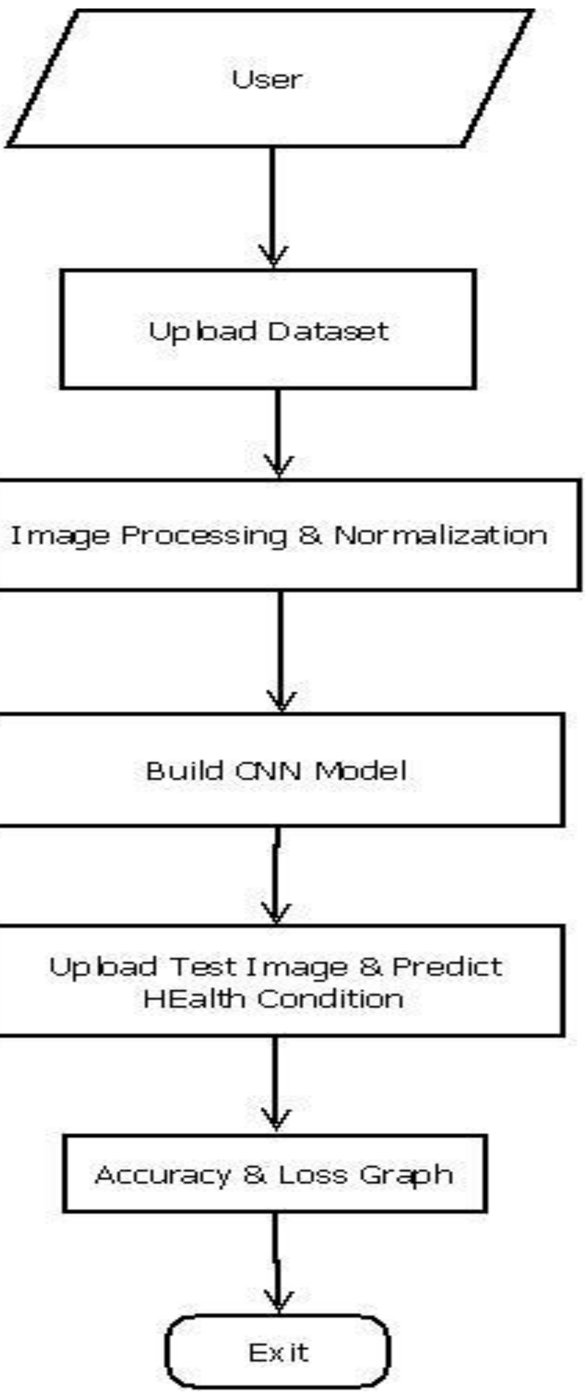

User
Up load Dataset
Image Processing & Normalization
Build CNN Model
Up load Test Image & Predict HEalth Condition
Accuracy & Loss Graph
Exit

Fig. 5.2. Data Flow Diagram

5.3 UML Diagrams

UML stands for Unified Modeling Language. UML is a general-purpose modeling language used in object-oriented software engineering. The standard was established by the Object Management Group.

Second, it is envisaged that the widespread use of object-oriented computer software models will result from the extensive use of UML. UML's present structure is mostly composed of a Meta-model and a Notation. Other methods and processes may be added in the near future to UML as well. Software system artifacts and other non-software system models can be specified, visualized, built and documented in a single vocabulary known as the Unified Modeling Language (UML).

- To build large, complex systems, the UML use a variety of tried-and-true engineering approaches.
- The UML is an essential part of object-oriented software development and the software development process. The UML is widely used to represent software project design in graphical notations.

GOALS:

The following are the primary aims of the UML design:

- To build large, complex systems, the UML use a variety of tried-and-true engineering approaches.
- The UML is an essential part of object-oriented software development and the software development process. The UML is widely used to represent software project design in graphical notations.
- The following are the primary aims of the UML design:
- Provide users with an expressive visual modeling Language so that they may create and trade meaningful models.
- Provide means for extending the fundamental concepts through extendibility and specialization.
- Programming languages and development processes should be able to stand on their own two feet.
- A formal framework for understanding the modeling language is provided in this step.
- Assist the expansion of the market for object-oriented (OO) tools.
- Assist with the implementation of higher-level development concepts such as teamwork and frameworks.
- Best practices should be incorporated.

5.3.1 Use Case Diagram

Diagrams in the Unified Modeling Language (UML) based on use cases are known as use cases in the UML. Use cases, actors, and any dependencies between them are all depicted graphically to give a clear picture of the system's capabilities. A use case diagram's primary goal is to explain how the system performs for each actor. System actors can be depicted by their responsibilities in the system.

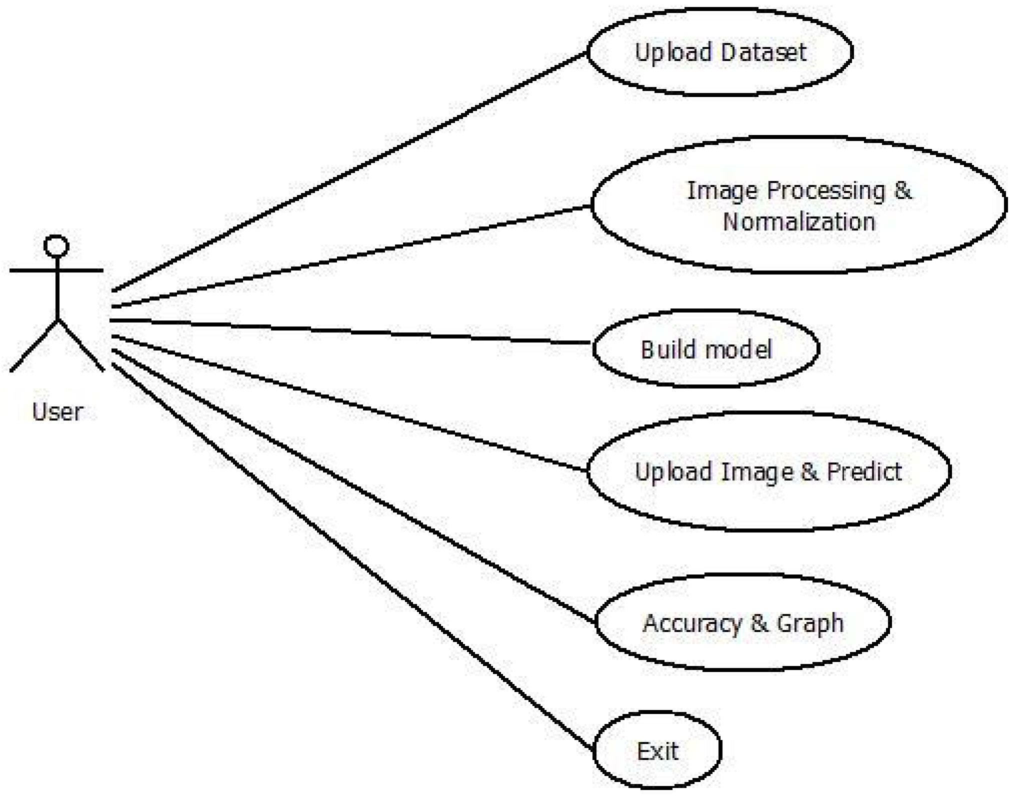


Fig. 5.3.1 Use Case Diagram

5.3.2 Class Diagram

Using the class diagram, you may fine-tune the use-case diagram and create a more precise blueprint for your system's architecture. As seen in the use case diagram, each actor is assigned to one of several classes, which are themselves subsets of the larger class diagram. Classes can have a “is-a” or a “has-a” relationship, depending on the context. Each class depicted in the diagram may be able to do a specific task. The class’s “methods” refer to these built-in features. There may also be “attributes” specific to each class that serve to differentiate them.

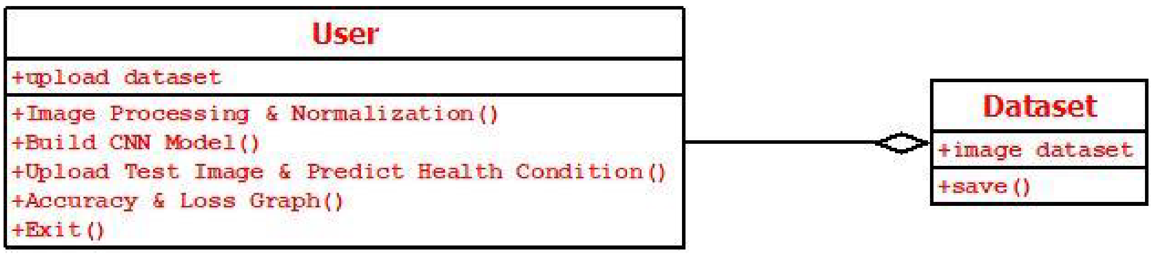


Fig. 5.3.2 Class Diagram

5.3.3 Object Diagram

Class diagrams have a variant known as the object diagram. A class is a type of object. An object is said to represent a class's status at a specific instant in the operating system. The object diagram depicts the current state of the system's various classes and their connections at a specific point in time.

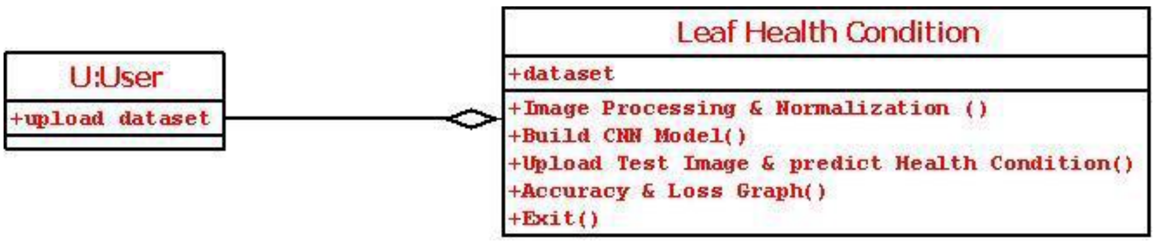


Fig. 5.3.3 Object Diagram

5.3.4 State Diagram

When you draw a state diagram, you're depicting all of the numerous states that various system objects can be in at a given time. When an event occurs, objects in the system alter their states. State diagrams are also used to show how an object's current state changes over time, in response to events in the system.

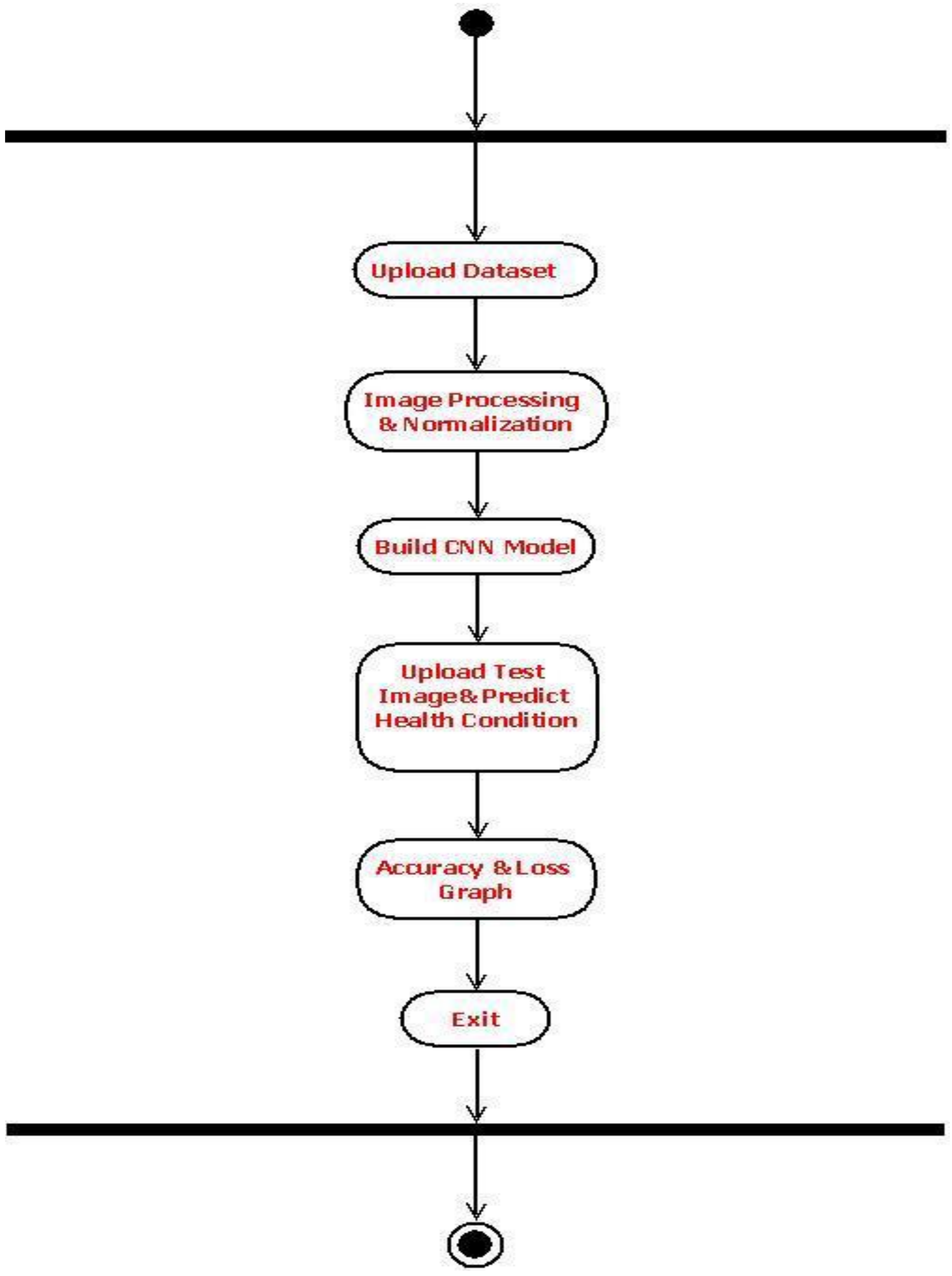

Upload Dataset
Image Processing & Normalization
Build CNN Model
Upload Test Image& Predict Health Condition
Accuracy & Loss Graph
Exit

Fig. 5.3.4 State Diagram

5.3.5 Activity Diagram

Using an activity diagram, you can see how the system works. There are similarities between an activity diagram and a state diagram in that they both comprise activities and actions as well as beginning and ending states as well as guard conditions.

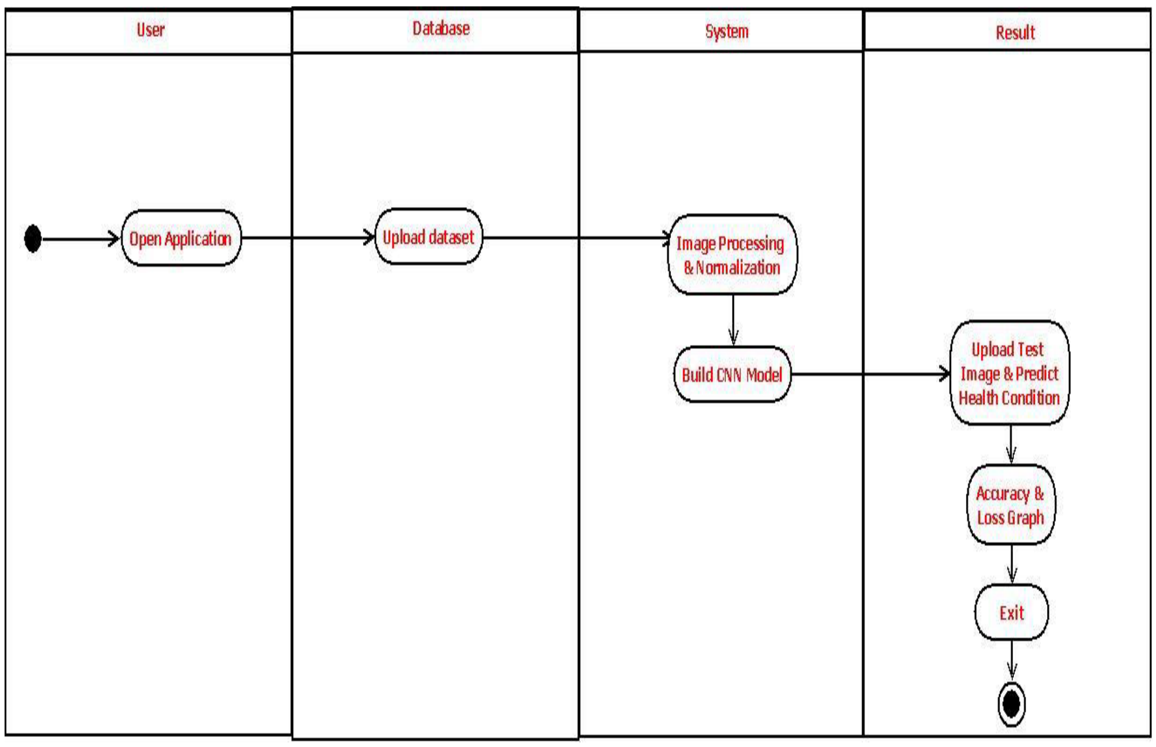


Fig. 5.3.5 Activity Diagram

5.3.6 Sequence Diagram

A sequence diagram is a visual representation of the system’s interactions. A sequence diagram’s most crucial feature is that it's time-ordered. In other words, the interactions between the items are represented step by step in this model. Messages are the means by which various elements of the sequence diagram communicate with one another.

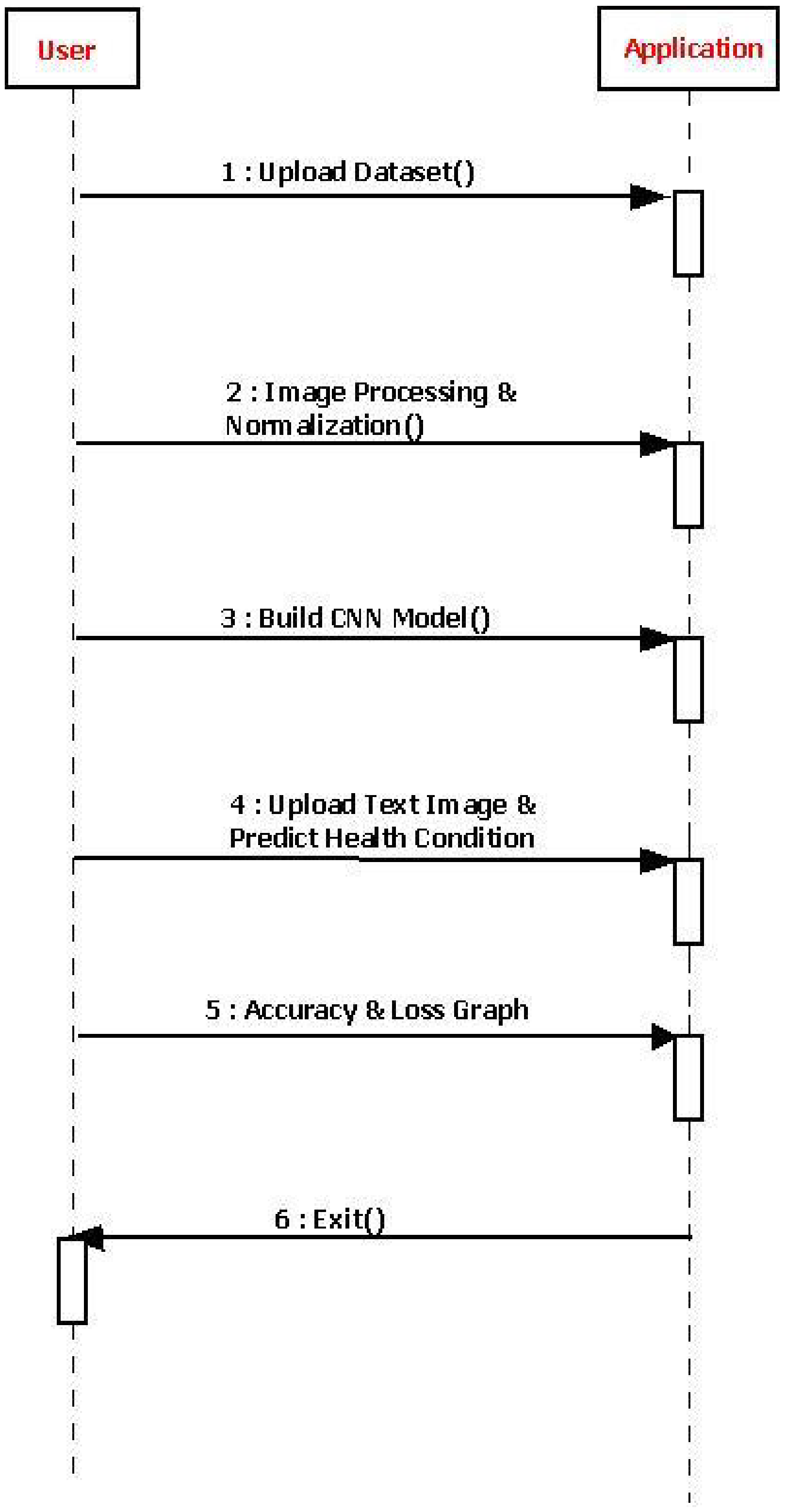

User
Application
1 : Upload Dataset()
2 : Image Processing & Normalization()
3 : Build CNN Model()
4 : Upload Text Image & Predict Health Condition
5 : Accuracy & Loss Graph
6 : Exit()

Fig. 5.3.6 Sequence Diagram

5.3.7 Collaboration Diagram

These relationships are depicted in a collaboration diagram. A numbering system aids in the tracking of the interactions order in the database. Using the cooperation diagram, it is possible to see all the ways in which one object can interact with the other.

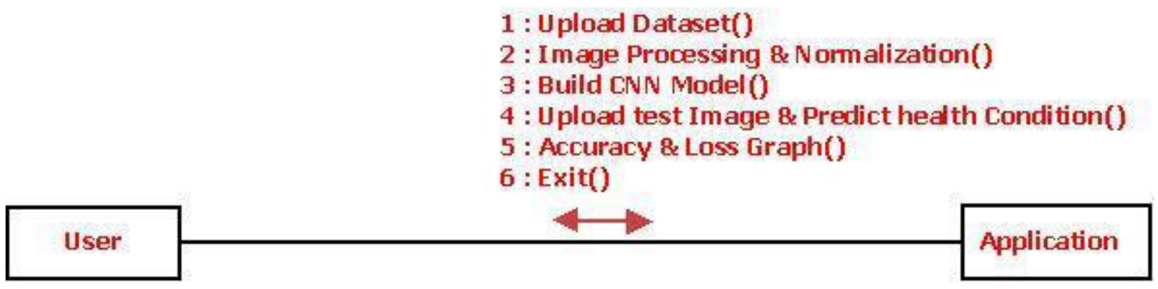

1 : Upload Dataset()
2 : Image Processing & Normalization()
3 : Build CNN Model()
4 : Upload test Image & Predict health Condition()
5 : Accuracy & Loss Graph()
6 : Exit()
User
Application

Fig. 5.3.7 Collaboration Diagram

CHAPTER-6

IMPLEMENTATION

6.1 Experimental Setup

A 64-bit Windows 10 PC was used in the experiment. The CNN model was built with Keras 2.2.4 deep learning framework, TensorFlow 1.14.0 backend, and Python 3.7.2.

6.2 Processing of Images

In addition to being taken in the field, all of the images were taken online. The dataset description includes images of Pepper_bell, Potato and Tomato. Image enhancement and retouching as a result of applying various enhancement techniques such as zoom, rotation, and horizontal and vertical shift to the collected photos with ImageDataGenerator in Keras, new images are generated at 224*224 pixel resolution.

6.3 CNN's Modeling School

The picture data set must be loaded in order to do training and testing. Class labels and images are kept in distinct arrays for training reasons. Due to the train-test split technique, training consumes 70% of the data, while testing consumes 30%. 30% of the data is utilized for validation and 70% of it is divided up again after that. The class labels are encoded as integers using a one-hot encoding procedure, and each label is represented as a vector rather than an integer.

Finally, the final fully linked layers are erased from Keras. Additions to the system that cannot be trained are made. To finish, we applied a softmax filter to the feature extractor's flattened output before applying the softmax filter[8,9]. Our model was designed from the ground up using the Adam optimizer with categorical cross entropy as the loss function for classification. Since the results remained stable after 10 epochs, we've stopped here. Fig. 6.3 depicts the classification procedure steps that we've taken. In support of the selected model, provide justification what we call "transfer learning" is the process through which what we learn in one setting can be applied to a different one.

Due to the fact that most real-world situations don't contain millions of labeled data points, transfer learning is extremely beneficial in training neural network models. In order to train a neural network from scratch, a large amount of data is required, however this data is not always available. Because the model has already been pre-trained, it is possible to build a robust machine learning model using only a small amount of training data. A ResNet50 that has been pre-trained on our dataset was utilised instead.

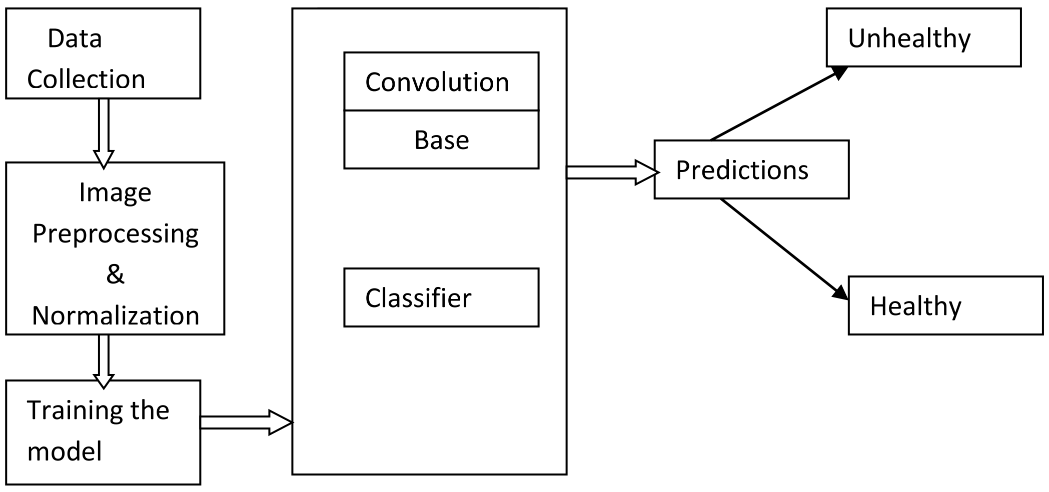


Fig. 6.3. Overview of the steps of the proposed model

6.4 CNN Technique

A deep learning technique known as Convolution Neural Networks is described in this study as an idea for automating government functions using Artificial Intelligence technology (CNN). People can read news and notifications about new government programs on the Internet, and they can subsequently express their opinions about the programs. This information can help the government make better judgments. Software like human brains are needed to identify public opinion about schemes automatically and this software must be able to discern whether the opinions expressed are in favour of positive or negative.

The suggestion is to construct a CNN model that works like a human brain in order to build such an automated opinion detecting system. In order to make this CNN model perform like automated decision making without any human interactions, we can generate it for any service and implement it. According to the concept of multiple models in which one model can identify or recognize human handwritten digits and the other model can detect sentiment from text sentences that can be offered by humans about government initiatives, this technique is already being proposed. We've included a new model in our extension that can identify emotion in a facial image as part of our extension model. Using facial expressions rather than words or sentences is a better way to convey feelings. So our new study can anticipate the emotions of people based on their faces.

As a way to explain how a convolutional neural network based image classifier works, we will develop a six-layer neural network that can detect and differentiate one image from the other. In order to run this network on a CPU, we'll need to design a very small network. In order to train a traditional neural network on a standard CPU, it would require a lot more parameters and take a long time. However, our goal is to demonstrate how to use Tensorflow to construct a real-world convolutional neural network.

Mathematical models for solving optimization problems are what neural networks are really all about. It is the basic unit of computation in neural networks, which is why they are formed of neurons. If an input (say, x) is fed into an active neuron, the neuron does some computations on it (like as multiplying it by a variable called w and then adding another one called b), the result of which is a new value (such as z=wx+b). Non-linear activation function f is used to produce the ultimate output of a neuron, which is called activation. Activation functions come in a variety of shapes and sizes. Sigmoid is a well-known activation function. The sigmoid neuron is the neuron that uses the sigmoid function as an activation function. Neurons are given names based on their activation roles, and there are numerous varieties to choose from, like RELU and TanH. This is the next building element of neural networks, and it is called a layer when neurons are stacked in one line. See the layered image below for more information[10].

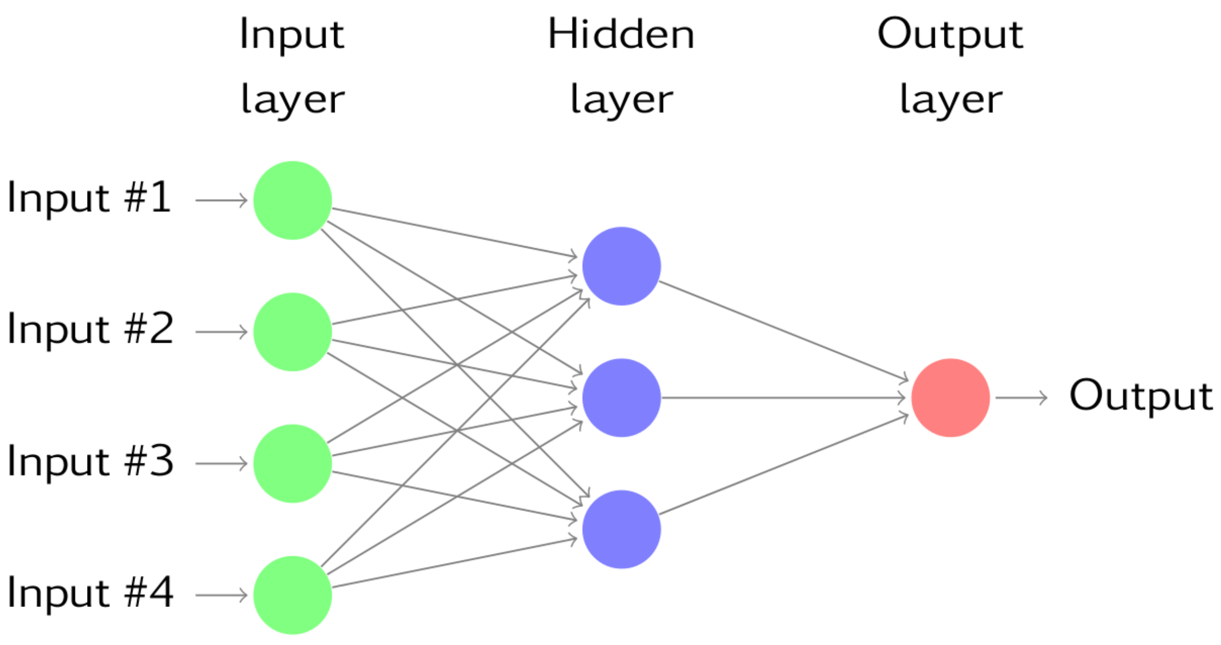


Fig. 6.4 Layered Image of CNN

CHAPTER-7
SOFTWARE ENVIRONMENT

7.1 Deep Neural Networks

Deep learning (sometimes known as "deep structured learning") is a machine learning method that is part of a broader family. It's possible to learn under close supervision, in a semi-unsupervised setting, or on one's own.

From computer vision and speech recognition to natural language processing and machine translation to bioinformatics, medication creation and medical image analysis, many different types of deep learning architectures, such as convolutional neural networks and recurrent neural networks, have been employed.

Artificial neural networks (ANNs) have been built based on the dispersed communication and information processing found in biological systems. There are numerous ways in which artificial neural networks (ANNs) differ from biological brains. Artificial neural networks are static and symbolic in comparison to the dynamic and analogue (plastic) brains of most organic animals.

In deep learning, there are several levels in the network. Non-polynomial activation functions with one hidden layer of unbounded breadth can be universal classifiers, according to early research. An infinite number of bounded-size layers, on the other hand, a recent version that allows for practical application and optimal implementation while maintaining theoretical universality under moderate conditions. These layers of deep learning can be heterogeneous and deviate from biologically-informed connectionist models because of this "organized" element. The goal of deep learning is to learn structured representations of large datasets layer by layer, which is accomplished by creating multiple layers of neurons in the brain.

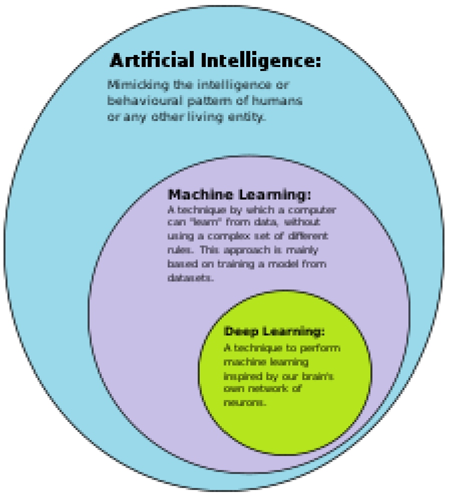

Artificial Intelligence:
Mimicking the intelligence or behavioural pattern of humans or any other living entity.
Machine Learning:
A technique by which a computer can "learn" from data, without using a complex set of different rules. This approach is mainly based on training a model from datasets.
Deep Learning:
A technique to perform machine learning inspired by our brain's own network of neurons.

Fig 7.1. Hierarchy

The network's architecture:

Models based on networks

Models of intelligence called "deep neural networks" were developed mathematically to resemble the functioning of the human brain.

• A network model is a way of describing how various layers of a network interact with one another.

• The following are some of the questions to consider when developing network models:

Other questions include: How many neurons should be used in each layer, as well as how the layers should be arranged? There are a number of typical CNN models that function well for a wide range of common issues. AlexNet, GoogleNet, Inception-ResNet, VGG, and so on are examples.

The frameworks for deep learning:

Because of its complexity, developing a deep learning solution is a major undertaking.

• A framework is a set of tools that makes it easier to create comprehensive learning solutions.

Using frameworks makes complex programming jobs easier.

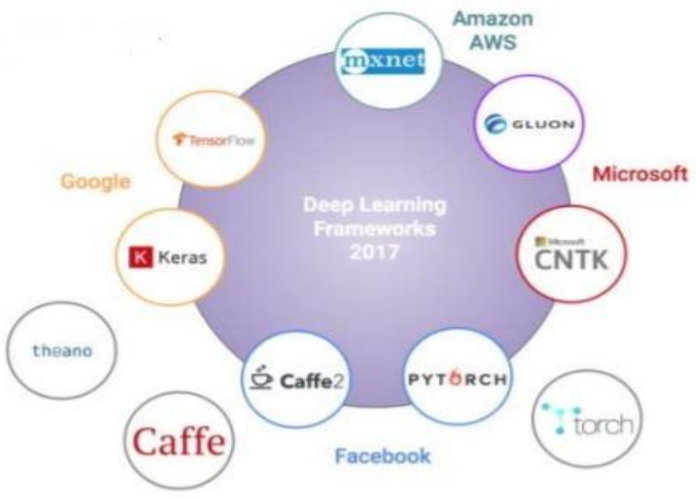


Fig. 7.1.1 Frameworks for Deep Learning

7.2 Popular Frameworks

Based on Github ratings and forks as well as Stack Overflow activity, TensorFlow is the most widely used deep learning framework.

For CNN modeling (imaging/computer vision applications), as well as its Model Zoo, Caffe was developed by the Berkeley Vision and Learning Center (BVLC) (a selection of pre-trained networks) All these frameworks are accompanied by interfaces that are wrapped around one or more frameworks. ' Keras is the most well-known and commonly used deep learning interface today. An API for deep learning built in Python, Keras is a high-level API.

Stream of deep learning

- A framework for development must be chosen before any work can begin.
- Choosing classes to train the network on from a labeled data set
- Making the first draught of a network diagram
- In addition, it is important to train the network.
- Creating a binary file with the parameters and architecture

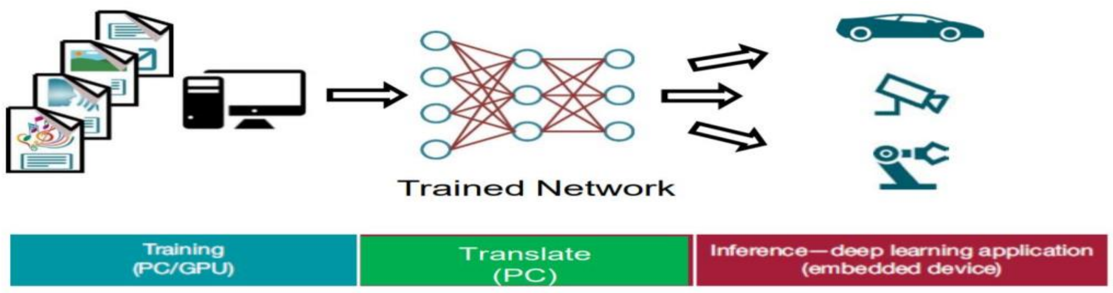


Fig. 7.2 Stream of Deep Learning

7.3 Training Data

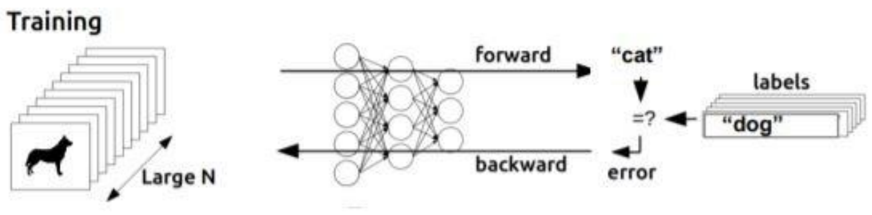


Fig. 7.3 Training Data

7.4 Introduction to Deep Learning Summary

• Exactly what is "deep learning"? Artificial intelligence (AI) is a catch-all term for any computer software that performs intelligently (AI). Artificial Intelligence (AI) is a subset of machine learning, which is a subset of deep learning as well.

• Deep learning is based on neural networks.

• Pattern recognition is the goal of neural networks, which are algorithms loosely built after the human brain.

• Deep learning: network types, nodes, layers, frameworks, and network models.

• The process of developing deep learning solutions

• Application areas

Sparse interactions, parameter sharing, and analogous representations are all key concepts in convolutional learning. As a further benefit, convolution allows for working with inputs of any size. Matrix multiplication by a matrix of parameters with a different parameter for each input unit and each output unit is used in traditional neural network layers. To put it another way, every output unit has a direct connection to every input unit. Reduce the number of parameters in the model to save memory while also increasing the model's statistical performance. As a result, fewer processes are required to calculate the final result. Efficiency gains of this magnitude are not uncommon. To get good results in many real applications, it is possible to store several orders of magnitude less data.

7.5 Deep Learning's Future

Scaling up deep learning systems often results in significant gains in performance. They generally do much better with a lot more data and a lot more computation. When compared to the number of synapses in the human brain, the GPT-318 language model has 175 billion parameters, but it produces considerably better text than the GPT-2 model, which has just 1.5 billion parameters. Meena and BlenderBot, two chatbots, keep getting better as they grow in popularity. Massive work is now being put into scaling up, which will greatly enhance present systems, but as has been discussed below, scaling cannot address the underlying flaws in current deep learning.

Several improvements to artificial intelligence can be identified by comparing human learning abilities to those of present AI.

1. Model-free reinforcement learning needs a lot more trial and error in comparison to supervised training. It seems that humans are capable of generalizing fairly well, despite their lack of experience. Whereas human beings need only a few examples to quickly adapt to changes in distribution, present systems are less resilient.Current deep learning techniques excel at problems involving perception and what is known as system . Deep learning is a viable solution for system

2. Operations that require a lot of planning where the problem lies. It has been the subject of machine learning theory since its conception because the id assumption argues that test examples are expected to come from the same distribution as training examples. Many agents' behaviors and a learning agent's ever-expanding horizon mean that assuming the universe is static in reality isn't feasible. The performance of

AI systems degrades as soon as they leave the lab and enter the real world. A subset of the more general goal of reducing sample complexity is to achieve greater robustness (also known as "out-of-distribution generalization") when confronted with a new task, like transfer learning or lifelong learning, or even just a shift in distribution or the relationship between world states and rewards (the number of examples needed to generalize well). For model-free reinforcement learning, each rewarded trial provides less information about the task than each tagged example, which is required by existing supervised learning systems. Other than the usual id-generalization, many studies show that humans can correctly interpret novel combinations of existing concepts, even if those combinations are extremely unlikely according to our training distribution, provided they respect high-level syntactic and semantic patterns we've already learned. Comparing neural network topologies has helped to clarify this ability to generalize.

7.6 Introduction to Python:

ABC was a precursor to Python, and Python is the successor to Python. In the year 1991, Guido Van Rossum came up with the idea. Code in this language is easy to understand thanks to the large amount of whitespace. It is an interpretive, general-purpose, and high-level programming language. In order to enable a variety of programming paradigms, an object-oriented approach was used. Python is a dynamically typed and garbage-collected programming language that supports both procedural and functional programming. This was followed by the release of Python 2.0 in the year 2000. Python 3.0 was released in late 2008.

Python's formatting was clean and the English keywords were used more frequently. The visual structure of a program typically depicts the program's semantic structure. The whitespace indentation is used instead of curly brackets and semicolons to delimit blocks in this language. The indentation parameters for the blocks are as follows:

Indentation will rise when certain statements are made.

This indicates the end of the current block by reducing the indentation.

Statements:

There is no need to declare the data type when assigning a value to a variable. The allocation of storage to names and objects can be caused by assigning a common value to several variables in succession..

The if statement, along with the else block and the elif (i.e., else if) block, will surely be executed.

In general, "import" statements are used to import modules whose variables and functions can be utilized in the current application. There are three ways to specify this:

Import module as [formalname].

from a module's *import* function

Invoke functions 1 and 2 from a module, respectively, as shown in Listing
Expressions:

The arithmetic operations are identical to those in other programming languages, with the exception of division, which is different. There are two basic sorts of divisions in Python:

i. The floor division (//) is

Division by a floating point number

The NumPy libraries utilise an infix operator that is denoted by @ for matrix multiplication starting with python 3.5.

Concatenation of tuples and string format are handled by + and percent, respectively, in Python.

Note:

** is the operator used for exponentiation in this example.

It is possible to use () as a key for dictionaries because tuples are immutable.

It is impossible to use a list as a dictionary key because lists are mutable and can be denoted by the symbol [].

Each element of the tuple, list, or array is referred to as a "shallow copy" when it is returned by Slice(:). The start index will be used for the slice, but the stop index will be omitted.

7.7 Libraries

The following ten libraries are the most frequently cited by machine learning programmers for inclusion in their work:

• TensorFlow

• Pandas

• Scikit-Learn

• NumPy

• Keras

• Pytorch

• LightGBM

• Eli5

• SciPy

• Theano

7.7.1 Numpy

"Numerical Python" encompasses linear algebra, Fourier transforms, and other fundamental concepts. An array containing real numbers can be used to represent images and binary row streams in this library's Array interface. The following attributes are included:

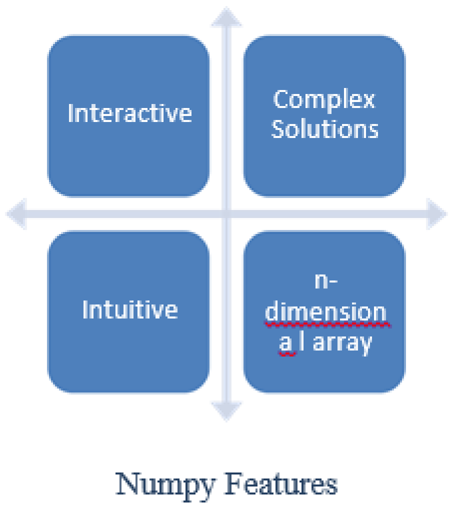


Fig. 7.7.1 Numpy Features

7.7.2 Scikit-Learn

In general, a python library for dealing with complicated data is linked to Numpy and SciPy. This software is used mostly for classification, regression, clustering, and dimensionality reduction. To import modules from it, the name "sklearn" is given. The following attributes are included:

Cross Validation
Unsupervised Learning
Feature extractio
Complex Calculations
Scikit Features

Fig. 7.7.2 Scikit Features

7.7.3 Pandas

For the most part, this library focuses on providing high-level data structures and a number of tools for evaluating them. It is responsible for the processing and modification of structured data. Executed operations are described here.

Slicing, Data Munging, Concatenation, Changing the index, Changing the column headers. The features are:

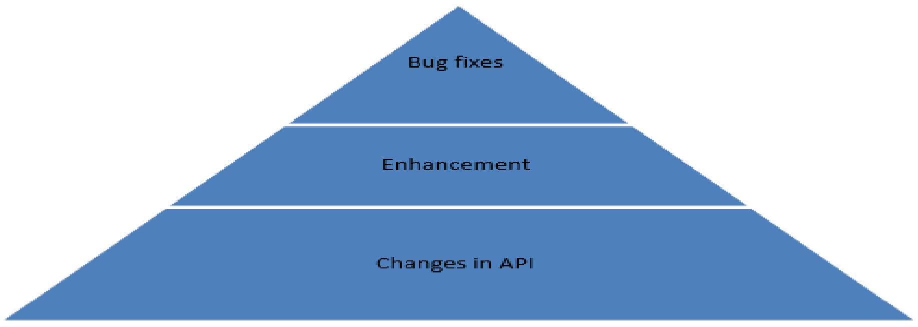


Fig. 7.7.3 Pandas

7.7.4 Matplotlib

Python programmers can make use of this module to create 2D graphics. Useful in python, shell, web servers and other GUI toolkits as well. There are two significant drawbacks to this module: it relies heavily on other packages, and it only works with Python. In terms of plots, there are three categories:

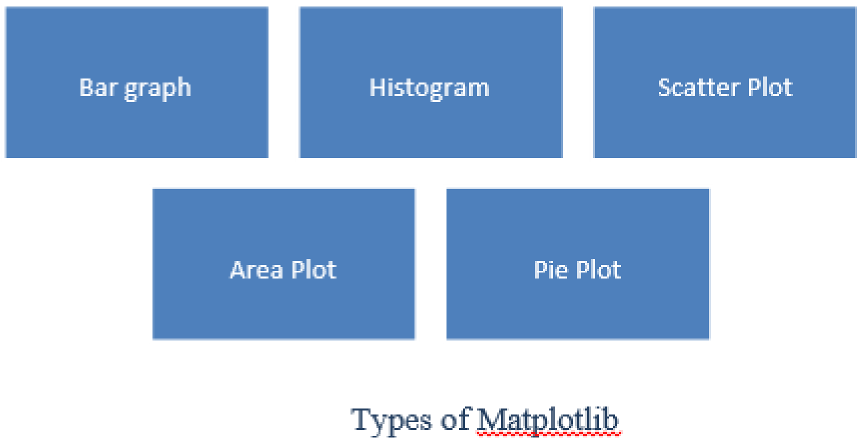
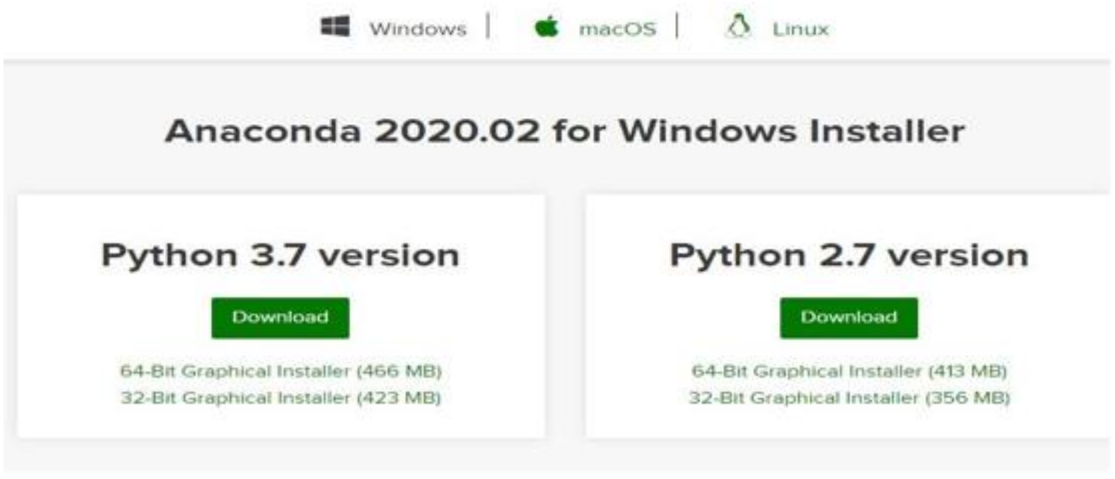


Fig. 7.7.4 Matplotlib

7.8 Anaconda(A Python Distribution):

7.8.1 Installation of Anaconda

Step 1: Go to anaconda navigator website for downloading.



Anaconda 2020.02 for Windows Installer

Python 3.7 version

64-Bit Graphical Installer (466 MB)
32-Bit Graphical Installer (423 MB)

Python 2.7 version

64-Bit Graphical Installer (413 MB)
32-Bit Graphical Installer (356 MB)

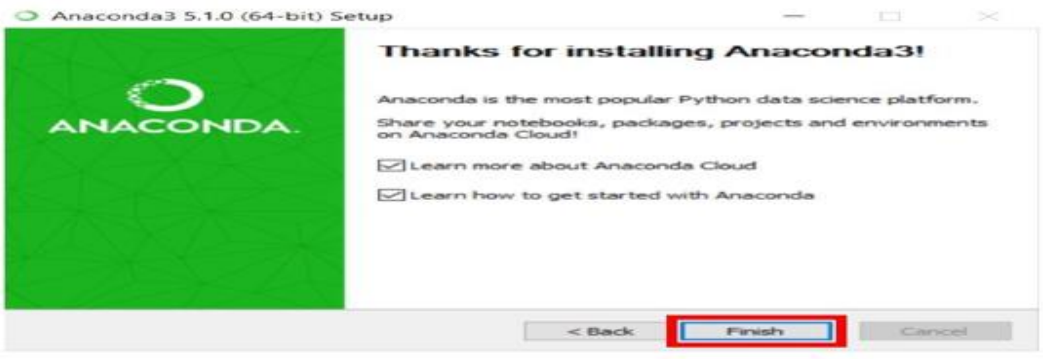
Fig. 7.8.1 Installation of Anaconda

7.8.2 Anaconda Navigator Installer

Step 2: Click on Anaconda3 that is available in Downloads.

Step 3: Click on Next and then accept the terms and conditions. After which a popup will come to select the path and the options will displayed for installation.

Step 4: Click on install button and hence the following screens will be displayed.



Thanks for installing Anaconda3!

Anaconda is the most popular Python data science platform.

Share your notebooks, packages, projects and environments on Anaconda Cloud!

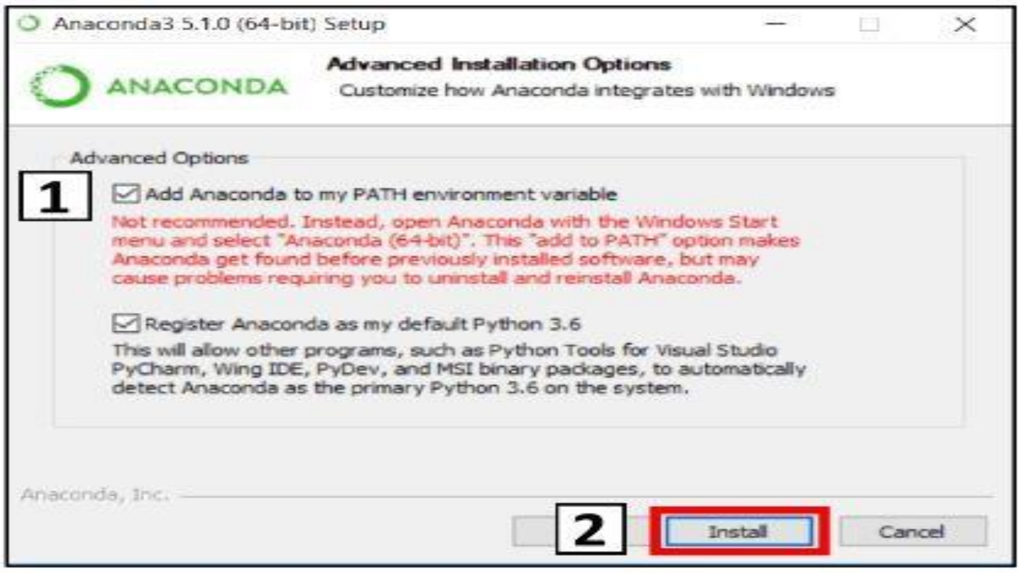
☑ Learn more about Anaconda Cloud

☑ Learn how to get started with Anaconda

Installation Successful

Fig. 7.8.2. Anaconda Navigator Installer

Installation Options


Anaconda3 5.1.0 (64-bit) Setup

Advanced Installation Options
Customize how Anaconda integrates with Windows

Advanced Options

1 [✓] Add Anaconda to my PATH environment variable

Not recommended. Instead, open Anaconda with the Windows Start menu and select "Anaconda (64-bit)". This "add to PATH" option makes Anaconda get found before previously installed software, but may cause problems requiring you to uninstall and reinstall Anaconda.

[✓] Register Anaconda as my default Python 3.6

This will allow other programs, such as Python Tools for Visual Studio PyCharm, Wing IDE, PyDev, and MSI binary packages, to automatically detect Anaconda as the primary Python 3.6 on the system.

Anaconda, Inc.

Fig. 7.8.3 Installation Options

Introduction:

For scientific computations, this is a free and open source distribution for R and Python computer languages aimed at streamlining the handling of packages and their deployment. The "conda" system, a package management system, is in charge of keeping track of the different versions. For MacOS, Linux, and Windows-based systems, this download comprises 1500 data science packages that were picked from the PyPI virtual environment manager and conda. You can use pip (recommended installation program) or one of the other two types of managers.

conda:

This manager will thoroughly examine the existing environment, which includes all previously installed and currently running packages, before adding the one that's now missing. Installing conda is as simple as typing conda install.

CHAPTER-8
SYSTEM TEST

In order to discover errors, testing is important. In order to uncover all conceivable flaws or vulnerabilities in a product, testing is the process of looking for them. Checking the functionality of parts, subassemblies, assemblies, and/or a finished product can be done using this approach Exercises are performed to ensure that the software system satisfies its specifications and user expectations, and does not fail in an unacceptable manner. There are many different sorts of tests. Each form of test is designed to suit a certain testing need.

8.1 What Tests Are Available?

8.1.1 Unit Tests

To guarantee that a program's fundamental logic is working effectively and that program inputs create legitimate outputs, unit testing is utilized. Validate all decision paths and internal code flow. It is the testing of the application's individual software units. It is done prior to the integration of a single unit. This is an invasive structural test that demands prior understanding of the system's architecture. This sort of test is intended to evaluate a specific business process, application, and/or system configuration at the component level. There are a number of approaches to ensure that a business process adheres to its written criteria, including unit tests.

8.1.2 Testing of the System as a Whole

The purpose of an integration test is to verify if two or more pieces of software can actually work together as a single unit. An event-driven method to testing focuses more on the basic outputs of screens and fields. Integration tests indicate that, despite the fact that the components were satisfied individually, the combination of components is proper and consistent. The goal of integration testing is to discover any difficulties that may occur as a result of merging distinct components.

8.1.3 A Test of Usefulness

Systems documentation, business and technical requirements, and user guides all play a role in functional testing since they demonstrate that the functionalities being tested are in fact present and accounted for.

Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classed of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classed of application outputs must be exercised
Systems/Procedures	: interfacing systems or procedures must be invoked.

Based on the requirements, critical functionalities, or specific test cases that need to be tested, functional tests are organized and created in accordance with these needs. The system's capacity to cover all important data fields, preset processes, and subsequent operations must also be verified. Before functional testing is finished, new tests are discovered and the value of the ones that already exist is assessed.

8.1.4 Verification of Input/Output

An integrated software system must meet a set of requirements before it can be tested to ensure it does. By putting a setup to the test, it may verify that it will provide the expected results. It is an example of systems testing to do a configuration-oriented system integration test. System testing is based on flow diagrams and explanations of how processes work, and the focus is on pre-driven process links and integration points.

8.2 Performing Experiments in A Black Box

Testers who are familiar with the program or its intended use conduct this type of testing. It's got a good point. This is the ideal tool for testing sites that can't be tested from the black box.

8.2.1 Testing in the Absence of a Test Environment

Software is tested using Black Box Testing, which means testers aren't privy to the module's internal structure, programming language, or workings. A definitive source document, such as a specification or requirements document, must be used to write black box tests, as with most other types of testing. Testing in which the program being tested is considered as a black box is called "black box testing." You can't look inside it. The test gives inputs and responds to outputs without taking into account the software's functionality.

8.2.2 Testing of the System as a Whole

As part of the software development lifecycle, unit testing is often performed in conjunction with coding and unit testing. Alternatively, unit testing can be performed as a separate process. Techniques for doing testing detailed functional tests are being prepared as well as manual field testing.

Goals of the exam

• Each and every field entry must function correctly.

• Functions can only be accessed by clicking on the button.

This includes the function execution, messages, and responses.

Aspects that will be examined

Inspect the data to make sure it is in the correct format.

Integrity Verification:

Integrating software components one at a time to look for bugs in the interfaces that could lead to errors is known as software integration testing.

Testing the interoperability of software components or applications, such as those in a software system or at the enterprise level, is the goal of an integration test.

The Process of Performing Acceptance Tests

Acceptance of the user any project's testing phase necessitates substantial input from the project's end users. It also ensures that the system's functional requirements are met, as well.

Results:

- In unit tests, all the decision paths and internal code flow have been validated successfully.
- In the integration testing, combination of components was identified to be proper and consistent.
- Each of the above-mentioned test cases was a complete success. There were no issues.
- The accuracy of the tested model is 98.7%.

CHAPTER-9

SCREENSHOTS

In this project we are applying deep learning convolution neural network (CNN) to predict crop disease and its pests to reduce economical loss in crop business. To build disease recognition model author is applying RESNET CNN model which consists of 3 parts

1) Feature Extraction: CNN compose of multiple layers and first layer define for feature extraction and this features will be extracted from given input image dataset or any other multidimensional dataset.
2) Feature Selection: Using this layer features will be selected by applying a layer called pooling or max polling.
3) Activation module: using this module RELU will be applied on input features to remove out unimportant features and hold only relevant important features
4) Flatten: This layer will be define to convert multidimensional input features into single dimensional input array
5) Dense: This layer can be used to connect one layer to other layer to receive input features from previous layer to new layer to further filter input features in next layer to get most important features from dataset to have best prediction result.

To implement this project we have used crop disease recognition dataset and this dataset saved inside ‘CropDiseaseDataset’ folder showing various type of healthy and unhealthy plants. There are 15 folders and each folder contains images of own leaf and in below screen you can see those image. You too just go inside any above folder to see images.

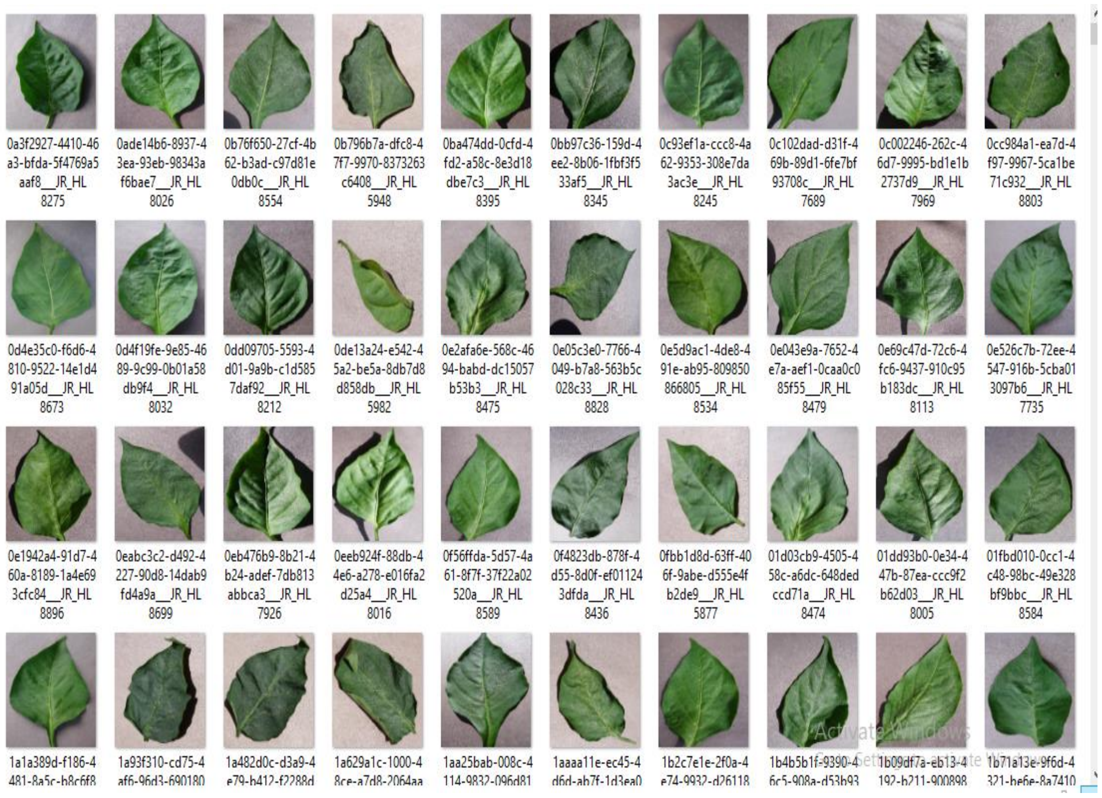

0a3f2927-4410-46a3-bfda-5f4769a5aaf8__JR_HL 8275
0ade14b6-8937-43ea-93eb-98343af6bae7__JR_HL 8026
0b76f650-27cf-4b62-b3ad-c97d81e0db0c__JR_HL 8554
0b796b7a-dfc8-47f7-9970-8373263c6408__JR_HL 5948
0ba474dd-0cfd-4fd2-a58c-8e3d18dbe7c3__JR_HL 8395
0bb97c36-159d-4ee2-8b06-1fbf3f533af5__JR_HL 8345
0c93ef1a-ccc8-4a62-9353-308e7da3ac3e__JR_HL 8245
0c102dad-d31f-469b-89d1-6fe7bf93708c__JR_HL 7689
0c002246-262c-46d7-9995-bd1e1b2737d9__JR_HL 7969
0cc984a1-ea7d-4f97-9967-5ca1be71c932__JR_HL 8803
0d4e35c0-f6d6-4810-9522-14e1d491a05d__JR_HL 8673
0d4f19fe-9e85-4689-9c99-0b01a58db9f4__JR_HL 8032
0dd09705-5593-4d01-9a9b-c1d5857daf92__JR_HL 8212
0de13a24-e542-45a2-be5a-8db7d8d858db__JR_HL 5982
0e2afa6e-568c-4694-babd-dc15057b53b3__JR_HL 8475
0e05c3e0-7766-4049-b7a8-563b5c028c33__JR_HL 8828
0e5d9ac1-4de8-491e-ab95-809850866805__JR_HL 8534
0e043e9a-7652-4e7a-aef1-0caa0c085f55__JR_HL 8479
0e69c47d-72c6-4fc6-9437-910c95b183dc__JR_HL 8113
0e526c7b-72ee-4547-916b-5cba013097b6__JR_HL 7735
0e1942a4-91d7-460a-8189-1a4e693cfc84__JR_HL 8896
0eabc3c2-d492-4227-90d8-14dab9fd4a9a__JR_HL 8699
0eb476b9-8b21-4b24-adef-7db813abbca3__JR_HL 7926
0eeb924f-88db-44e6-a278-e016fa2d25a4__JR_HL 8016
0f56ffda-5d57-4a61-8f7f-37f22a02520a__JR_HL 8589
0f4823db-878f-4d55-8d0f-ef011243dfda__JR_HL 8436
0fbb1d8d-63ff-406f-9abe-d555e4fb2de9__JR_HL 5877
01d03cb9-4505-458c-a6dc-648dedccd71a__JR_HL 8474
01dd93b0-0e34-447b-87ea-ccc9f2b62d03__JR_HL 8005
01fbd010-0cc1-4c48-98bc-49e328bf9bbc__JR_HL 8584
1a1a389d-f186-4481-8a5c-b8c6f8
1a93f310-cd75-4af6-96d3-690180
1a482d0c-d3a9-4e79-b412-f2288d
1a629a1c-1000-48ce-a7d8-2064aa
1aa25bab-008c-4114-9832-096d81
1aaaa11e-ec45-4d6d-ab7f-1d3ea0
1b2c7e1e-2f0a-4e74-9932-d26118
1b4b5b1f-9390-46c5-908a-d53b93
1b09df7a-eb13-4192-b211-900898
1b71a13e-9f6d-4321-be6e-8a7410

Fig. 9.1 Dataset Images

To run project double click on ‘CropDisease.py’ file to get below screen

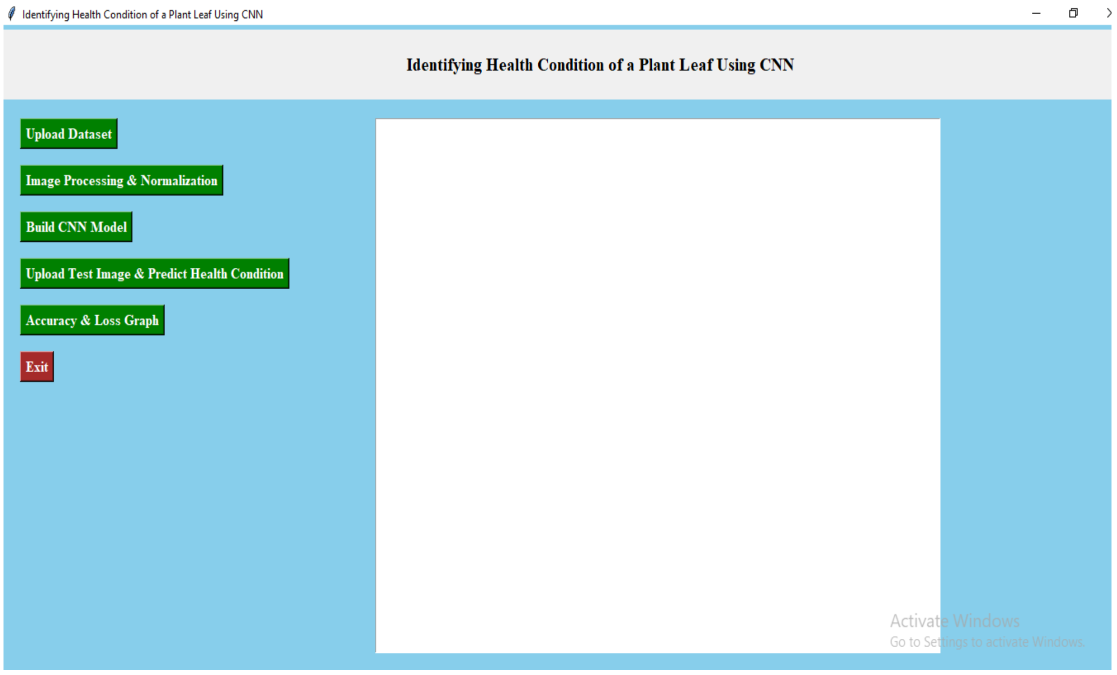
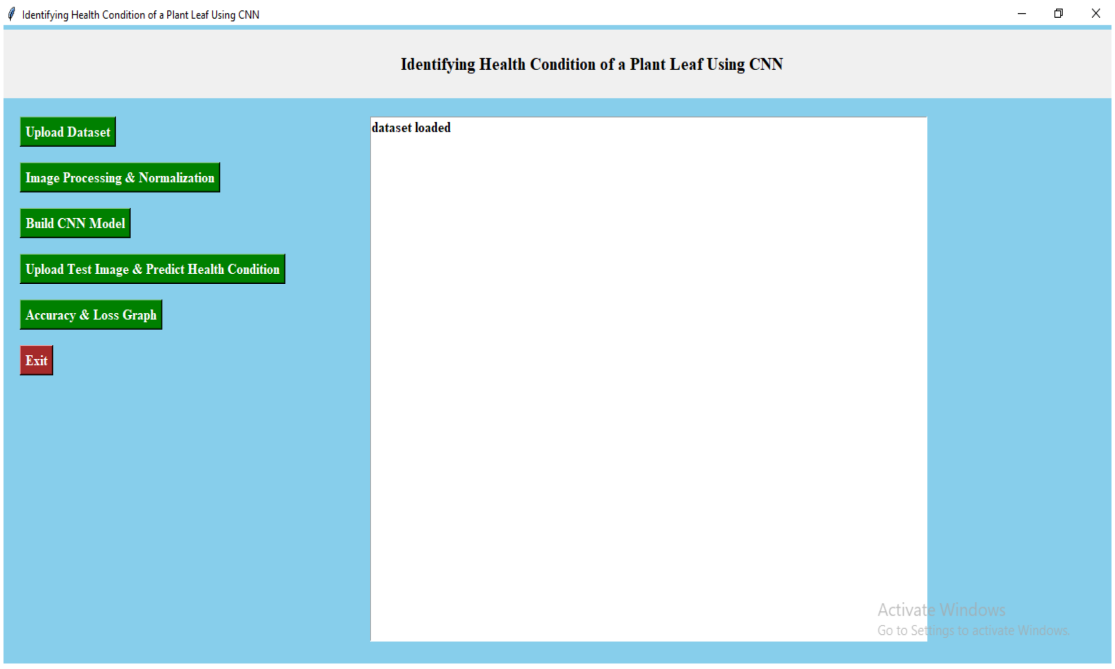

Identifying Health Condition of a Plant Leaf Using CNN
Identifying Health Condition of a Plant Leaf Using CNN
Upload Dataset
Image Processing & Normalization
Build CNN Model
Upload Test Image & Predict Health Condition
Accuracy & Loss Graph
Exit

Fig. 9.2 Python application

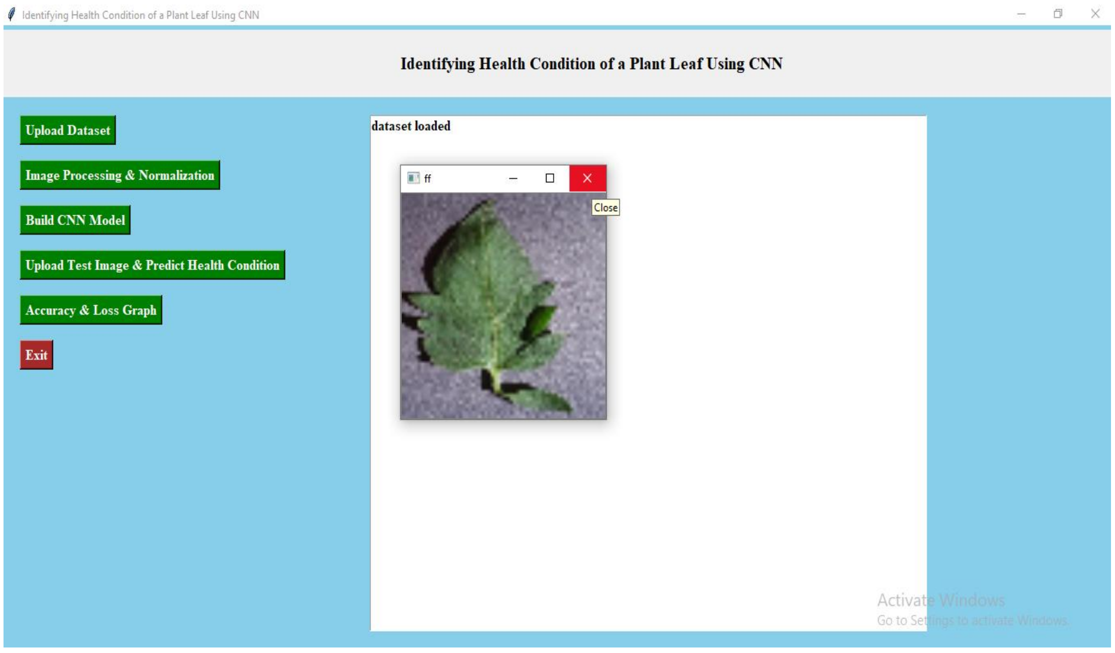
In above screen click on ‘Upload Dataset’ button to upload dataset images.


Identifying Health Condition of a Plant Leaf Using CNN
Identifying Health Condition of a Plant Leaf Using CNN
Upload Dataset
Image Processing & Normalization
Build CNN Model
Upload Test Image & Predict Health Condition
Accuracy & Loss Graph
Exit
Activate Windows
Go to Settings to activate Windows.

dataset loaded

Fig. 9.3 Upload Dataset

In above screen dataset loaded and now click on 'Image Processing & Normalization' button to read all images and then process images to normalize by converting each image pixel value between 0 and 1 and for that normalization we will divide image pixels with 255 and then get value as 0 or 1 as all images pixel value will be between 0 to 255


Identifying Health Condition of a Plant Leaf Using CNN
Identifying Health Condition of a Plant Leaf Using CNN
Upload Dataset
Image Processing & Normalization
Build CNN Model
Upload Test Image & Predict Health Condition
Accuracy & Loss Graph
Exit
ff
Close
Activate Windows
Go to Settings to activate Windows.

dataset loaded

Fig. 9.4 Image Processing & Normalization

In above screen after applying normalization we are just displaying one random image from dataset to check whether images loaded and process properly or not and now you close above image to get below screen.

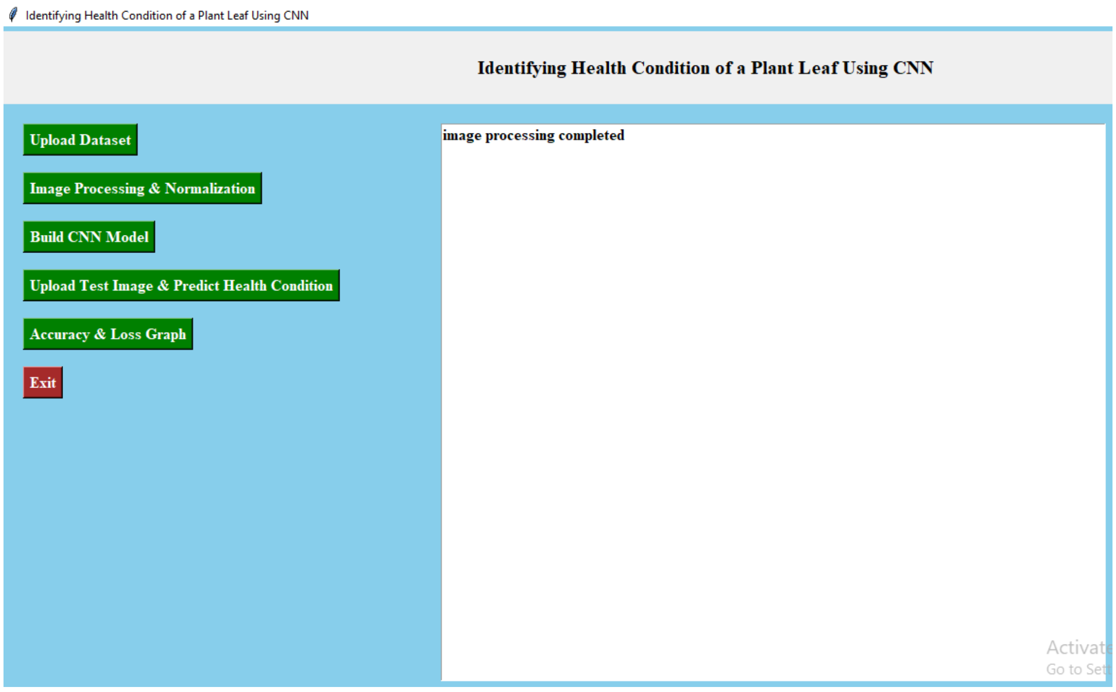
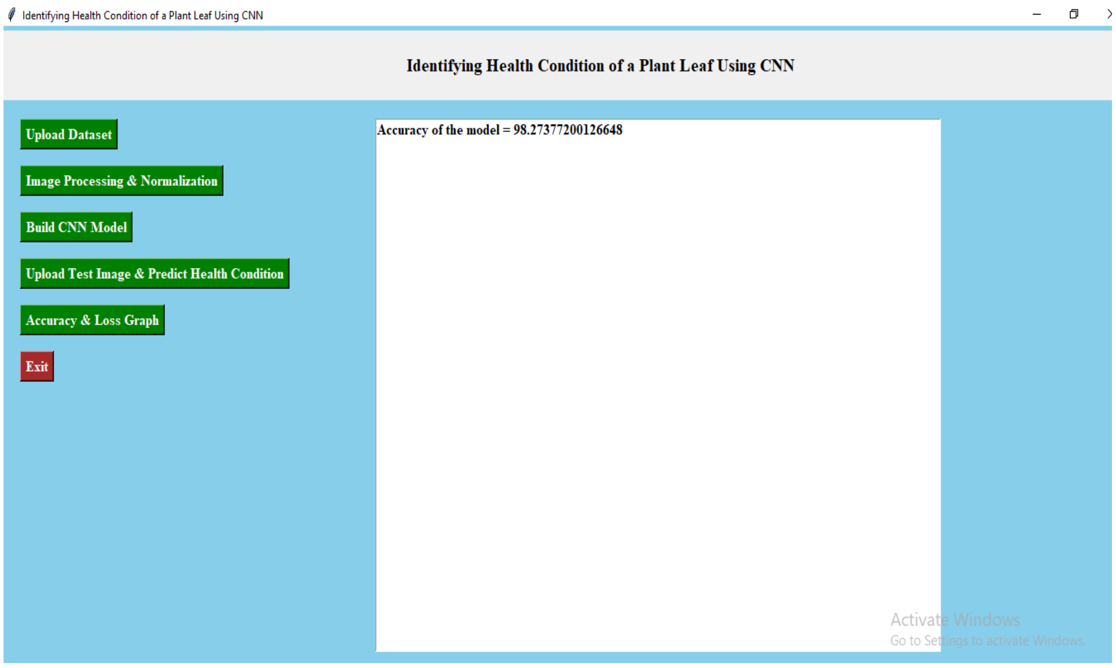

Identifying Health Condition of a Plant Leaf Using CNN
Identifying Health Condition of a Plant Leaf Using CNN
Upload Dataset
Image Processing & Normalization
Build CNN Model
Upload Test Image & Predict Health Condition
Accuracy & Loss Graph
Exit

image processing completed

Fig. 9.5 Completion of Image Processing

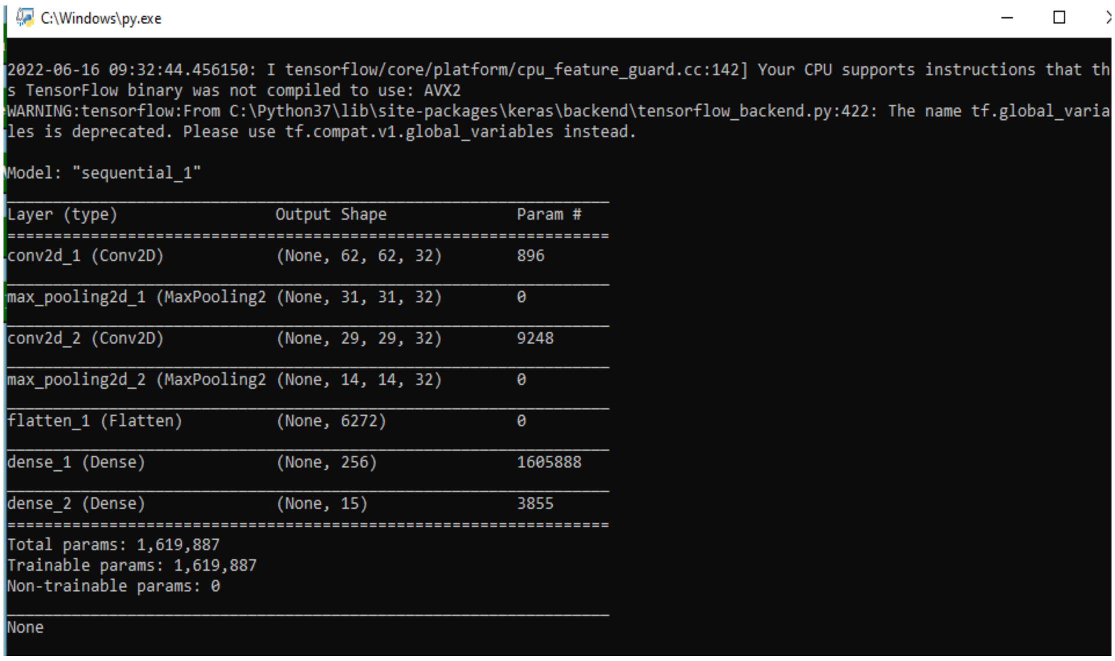
All images process successfully and now dataset images are ready and now click on ‘Build CNN Model’ button to build CNN model.


Identifying Health Condition of a Plant Leaf Using CNN
Identifying Health Condition of a Plant Leaf Using CNN
Upload Dataset
Image Processing & Normalization
Build CNN Model
Upload Test Image & Predict Health Condition
Accuracy & Loss Graph
Exit

Accuracy of the model = 98.27377200126648

Fig. 9.6 Build CNN Model

In above screen CNN model generated and its prediction accuracy is 98% and in below console screen we can see all CNN layers details.



```
C:\Windows\py.exe

2022-06-16 09:32:44.456150: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that th
s TensorFlow binary was not compiled to use: AVX2
WARNING:tensorflow:From C:\Python37\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.global_varia
les is deprecated. Please use tf.compat.v1.global_variables instead.

Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 62, 62, 32)        896
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 31, 31, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 29, 29, 32)        9248
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
flatten_1 (Flatten)          (None, 6272)              0
_________________________________________________________________
dense_1 (Dense)              (None, 256)               1605888
_________________________________________________________________
dense_2 (Dense)              (None, 15)                3855
=================================================================
Total params: 1,619,887
Trainable params: 1,619,887
Non-trainable params: 0
_________________________________________________________________
None
```

Fig. 9.7 Model Summary

In above screen we can see we have used CONV2D, MAXPOOLING, FLATTEN and DENSE layer to build crop disease recognition model and RELU details you can see in code. Now model is ready and now click on 'Upload Test Image & Predict Disease' button to upload any test image and then application will predict disease or healthy from that image.

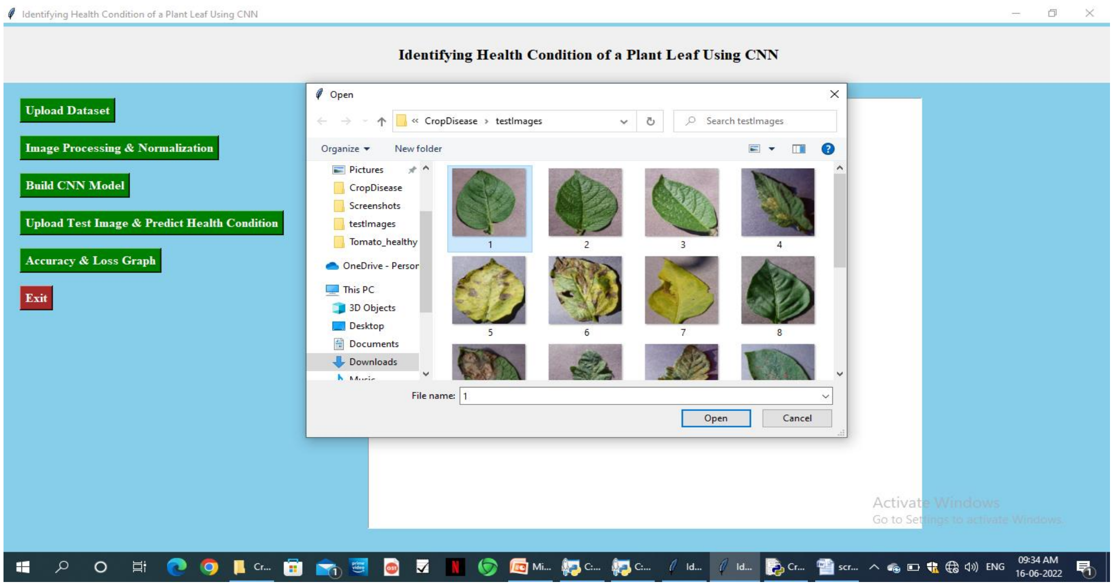


Fig. 9.8 Upload Plant Leaf

In above screen selecting and uploading '1.JPG' image file and then click on 'Open' button to get below prediction result.

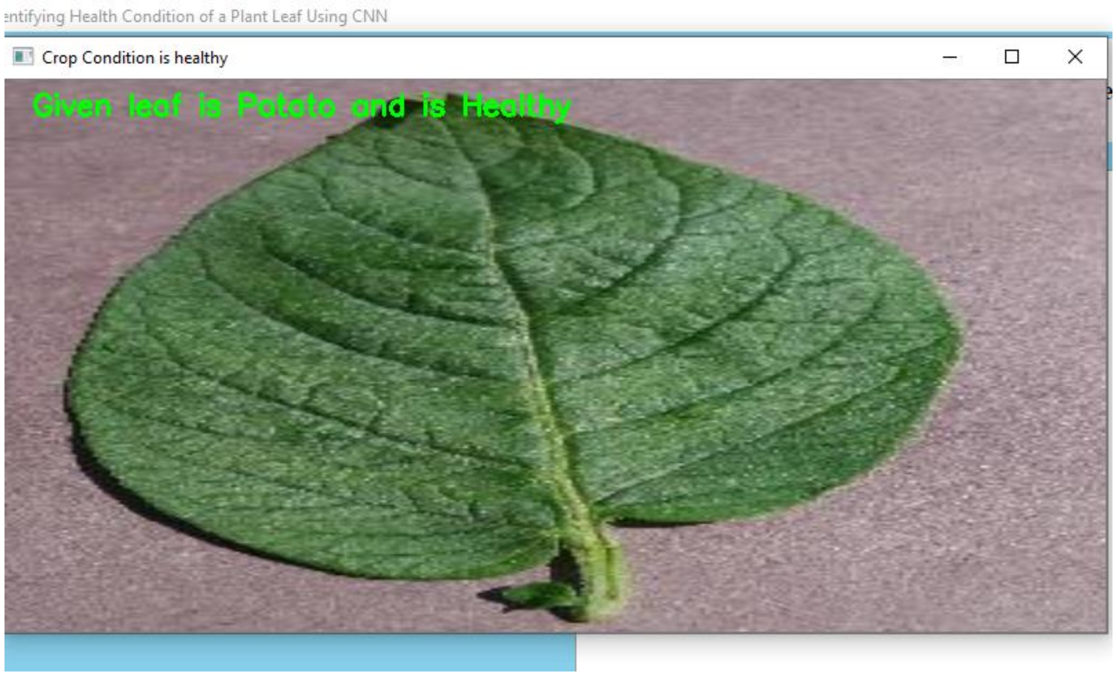

entifying Health Condition of a Plant Leaf Using CNN
Crop Condition is healthy
Given leaf is Potato and is Healthy

Fig. 9.9 Output 1

Selecting and uploading '7.JPG' image file and then click on 'Open' button to get below prediction result.

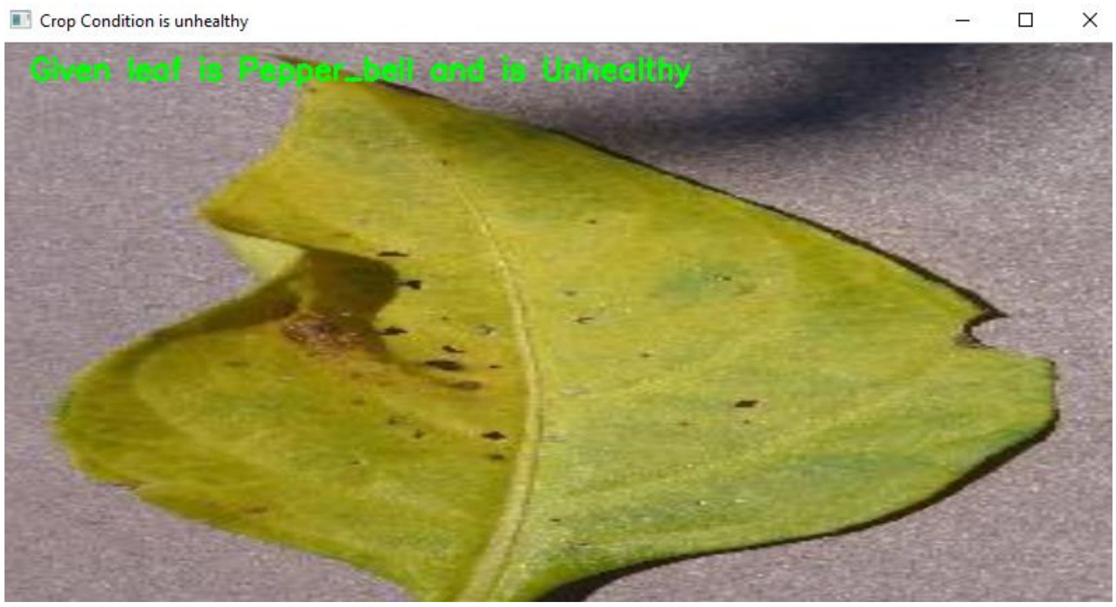

Crop Condition is unhealthy
Given leaf is Pepper_bell and is Unhealthy

Fig. 9.10 Ouput 2

CHAPTER-10

RESULT

10.1 Calculations

The suggested model properly predicted 98% of the trained data after running for 10 epochs on training data. A CNN model without transfer learning was trained and tested using the same data set, but with different split ratios. Optimization settings including batch size, epochs and Adam were fine-tuned. In the CNN model without transfer learning, ReLU, Maxpooling, and dropout layers are followed by two Fully Connected Layers and SoftMax. Fig 10.1 displays the CNN's training and validation accuracy, as well as the number of epochs, using Transfer Learning.

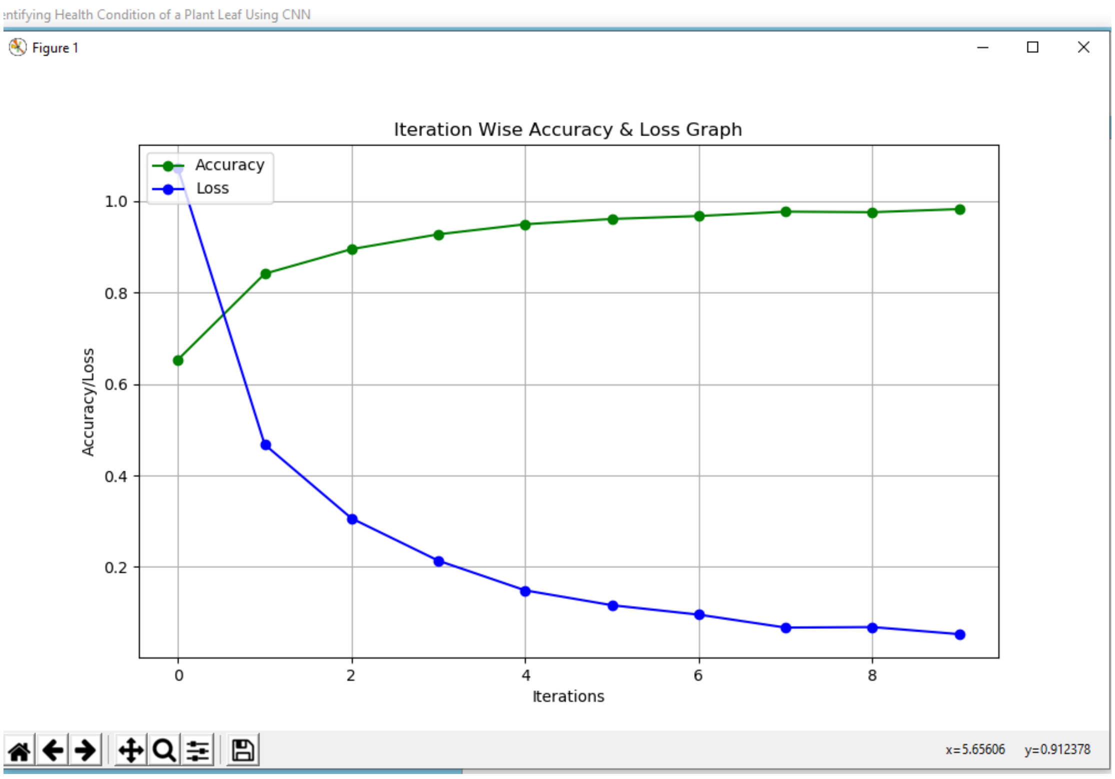


Fig. 10.1 Graph

In above graph x-axis represents epoch/iterations and y-axis represents accuracy/loss and green line represents accuracy and blue line represents loss and from above graph we can see with each increasing iteration accuracy is getting better and better and loss getting decrease

CONCLUSION

In this research, we present a deep learning architecture that classifies 98% of the test photos. As a result of fine-tuning the ResNet50 model, we were able to significantly increase the model's performance. We capped the number of epochs employed at 10 after receiving data showing no improvement in accuracy or decrease in loss on either the training or validation sets.

In the future:

More photos from agricultural areas and agricultural research organizations are needed in the future to further increase the accuracy of our results. In the future, we want to include a cross-validation mechanism to further verify our findings. It would also be beneficial if we could compare the findings achieved with those of more advanced deep learning models and other current efforts in progress. Other plant leaf health conditions, which are significant crops in India, can be detected using the model described here.

REFERENCES

[1] Iniyan, S. & Rethnaraj, Jebakumar & Poobalasubramanian, Mangalraj & Mohit, Mayank & Nanda, Aroop, “Plant Disease Identification and Detection Using Support Vector Machines and Artificial Neural Networks”, 10.1007/978-981-15-0199-9_2.

[2] Merugu Sai Charan1, Mohammed Abrar2, Bejjam Vasundhara Devi3, “Apple Leaf Diseases Classification Using CNN with Transfer Learning”, International Journal for Research in Applied Science & Engineering Technology (IJRASET), ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538, Volume 10 Issue VI June 2022.

[3] Sharada P. Mohanty, David P. Hughes and Marcel Salathe, “Using Deep Learning for Image-Based Plant Disease Detection”, Front Plant Sci. 2016 Sep 22;7:1419. doi: 10.3389/fpls.2016.01419. PMID: 27713752; PMCID: PMC5032846.

[4] P. B. Padol and A. A. Yadav, “SVM classifier based grape leaf disease detection,” Conference on Advances in Signal Processing (CASP), pp. 175-179, 2016.

[5] Simonyan, Karen & Zisserman, Andrew. (2014), “Very Deep Convolutional Networks for Large – Scale Image Recognition”, arXiv 1409.1556.

[6] S. Xie, T. Yang, Xiaoyu Wang and Yuanqing Lin, "Hyper-class augmented and regularized deep learning for fine-grained image classification," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2645-2654, doi: 10.1109/CVPR.2015.7298880.

[7] Nageshwar Jaiswal and Vivek Sarnaik, “Detection of Plant Leaf Disease Using CNN Algorithm”, ISSN (Online): 2320-9364, ISSN (Print): 2320-9356.

[8] Nagaveni B. Nimbal, Anushree S, G R Sahana, Madhu K V, Soundarya R, “Green leaf Disease Detection Using CNN”, International Journal of Engineering Reasearch & Technology, ISSN: 2278-0181.

[9] K. Simonyan, A. Zisserman, “Very Deep Convolutional Networks for LargeScale Image Recognition”, Arxiv preprint arXiv: arXiv,pp. 1409- 1556,2015.

[10] Lamba, Monika, Gigras, Yogita and Dhull, Anuradha, “Classification of plant diseases using machine and deep learning”,vol. 11, n0. 1, 2021, pp. 491-508.