

A PROJECT REPORT ON
INTEGRATING MACHINE LEARNING TECHNIQUES FOR PLANT DISEASE
DETECTION AND FRUIT CLASSIFICATION

Submitted In partial fulfilment of the requirements for the award of the degree of
BACHELOR OF TECHNOLOGY
IN
ELECTRONICS & COMMUNICATION ENGINEERING

SUBMITTED BY

R. SATYA SRI RANJANI	(20NE1A04E1)
SK. NAGULBI	(20NE1A04F8)
T. KIRAN KUMAR REDDY	(20NE1A04F9)
SK. ABDUL RAZAK	(21NE5A0416)

Under The Esteemed Guidance of

Guide

Mr. THATI JAGADEESH
M Tech, M.S, (Ph.D).
Associate Professor, Dept. of ECE

Co - Guide

DR. CH. VENKATA SIVA PRASAD
M Tech, Ph.D.
Professor



DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING
TIRUMALA ENGINEERING COLLEGE
(AUTONOMOUS)

An ISO 9001:2015 Certified Institution, Accredited by NAAC(A+) & NBA
(APPROVED BY AICTE, NEW DELHI & AFFILIATED TO JNTUK, KAKINADA)
NARASARAOPET-522601
2020-2024

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING
TIRUMALA ENGINEERING COLLEGE
(AUTONOMOUS)

An ISO 9001:2015 Certified Institution, Accredited by NAAC(A+) & NBA
(APPROVED BY AICTE, NEW DELHI & AFFILIATED TO JNTUK, KAKINADA)
NARASARAOPET-522601
2020-2024



CERTIFICATE

This is to certify that the project report entitled “Integrating Machine Learning Techniques for Plant disease detection and Fruit classification” is the bonafide work done by R. SATYA SRI RANJANI (20NE1A04E1), SK. NAGULBI (20NE1A04F8), T. KIRAN KUMAR REDDY (20NE1A04F9), SK. ABDUL RAZAK (21NE5A0416) submitted in partial fulfillment of the requirements for the award of Bachelor of Technology in Electronics Communication of Engineering by JNTU, Kakinada and is submitted in the Department of Electronics & Communication of Engineering, Tirumala Engineering College, Jonnalagadda, Narasaraopet.

Project Guide

Head of the Department

External Examiner

ACKNOWLEDGEMENT

A successful project is a fruitful culmination of efforts by many people, some were directly involved and some others quickly encouraged and supported from the background.

We would like to gratefully acknowledge our project Guide and Head of the department **Mr. Thati Jagadeesh M.Tech, M.S, (Ph.D), Associate Professor**, has been abundantly helpful and has assisted us in numerous ways. We specially thank her for infinite patience. The discussions we had with were in valuable.

We take immense pleasure in thanking of our beloved Principal, **Dr Y.V.Narayana B.Tech, ME, Ph.D., FIETE**, Tirumala Engineering College for having permitted us to carry out this project work.

We are grateful to our college management for providing us excellent lab facilities and inspiring us through their valuable messages.

Our sincere thanks to the teaching and non-teaching staff for their support and co- operation in fulfillment of this project. Our heartfelt thanks to our parents and friends for helping and encouraging us throughout the completion of the project.

Project Associates

R. SATYA SRI RANJANI (20NE1A04E1)

SK. NAGULBI (20NE1A04F8)

T. KIRAN KUMAR REDDY (20NE1A04F9)

SK. ABDUL RAZAK (21NE5A0416)

DECLARATION

We are the students of Tirumala Engineering College, **R. Satya Sri Ranjani (20NE1A04E1), Sk. Nagulbi (20NE1A04F8), T. Kiran Kumar Reddy (20NE1A04F9), Sk. Abdul Razak (21NE5A0416)** hereby declare that is project work entitled "**Integrating Machine Learning Techniques for Plant Disease Detection and Fruit Classification**" under the esteemed guidance of **Mr. Thati Jagadeesh M.Tech, M.S, (Ph.D)** Department of **Electronics & Communication Engineering** is submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in **Electronics and Communication Engineering**.

This is a Record of bonafide work carried out by us and the results embodied in this Project Report have not been reproduced or copied from any source. The results embodied in this Project Report have not submitted to any other University or Institute for the Award of any other degree or diploma.

Project Associates

R. SATYA SRI RANJANI	(20NE1A04E1)
SK. NAGULBI	(20NE1A04F8)
T. KIRAN KUMAR REDDY	(20NE1A04F9)
SK. ABDUL RAZAK	(21NE5A0416)

TABLE OF CONTENTS

	Page No
List of Figures	
List of Tables	
Abstract	
Chapter-1: INTRODUCTION	1-7
1.1 Introduction to the project	2-3
1.2 Problem Definition	4
1.3 Area of Project	4
1.4 Goals	4
1.5 Tools Required	5-7
1.6 Organization of the project	7
Chapter-2: BACKGROUND	8-16
2.1 Overview of Technologies used	9-14
2.1.1 Convolutional Neural Networks	9-10
2.1.2 Region-based Convolutional Neural Networks	10-11
2.1.3 Fast Region-based Convolutional Neural Network	11-12
2.1.4 Technologies for Model Training and Deployment	12
2.1.5 Web Development Technologies for User Interface	13
2.1.6 Image Processing Libraries and Tools	13
2.1.7 Data Management Technologies	13

2.1.8 Cloud Computing and Scalability	13
2.1.9 Ethical and Legal Considerations	14
2.2 Python	14-16
2.2.1 General Overview	14
2.2.2 Syntax and Structure	14
2.2.3 Features	14-15
2.2.4 Applications	15
2.2.5 Tools and Ecosystem	15
2.2.6 Recent Developments	16
Chapter-3: PROPOSED METHOD	17-29
3.1 Proposed System	19
3.2 Key advantages of Proposed System	20
3.3 Block Diagram of Faster R-CNN	21-29
3.3.1 Explanation	22-24
3.3.2 Region Proposal Network (RPN)	25-26
3.3.3.Region of Interest (RoI) Pooling Layer	27-28
3.3.4 Advantages	29
3.3.5 Applications	29
Chapter-4: SOFTWARE TESTING	30-38
4.1 Dataset	31-32
4.2 Testing	33-38

Chapter-5: RESULTS	39-46
5.1 Plant Disease Detection results	40-42
5.1.1 Qualitative results for plant disease detection	40
5.1.2 Quantitative results for plant disease detection	41-42
5.2 Fruit Classification results	43-46
5.2.1 Qualitative results for fruit classification	43-44
5.2.2 Quantitative results for fruit classification	45-46
Chapter-6: CONCLUSION	47-48
REFERENCES	49-52

LIST OF FIGURES

Name of the Figure	Page No
Fig-2.1. Schematic Diagram for CNN	9
Fig-2.2. Schematic Diagram for R-CNN	10
Fig-2.3. Fast R-CNN Architecture	12
Fig-2.4. Python 3	16
Fig-3.1 Block Diagram of Faster R-CNN	21
Fig-3.2 Faster R-CNN Architecture	22
Fig-3.3 Region Proposal Network (RPN)	25
Fig-3.4 Region of Interest (ROI) pooling	27
Fig-4.1 Dataset for plant disease detection	31
Fig-4.2 Dataset for fruit Classification	32
Fig-5.1 Feature Visualization	40
Fig-5.2 Accuracy for plant disease detection	42
Fig-5.3 Top 20 fruits by count	44
Fig-5.4 Feature Visualization of fruits	44
Fig-5.5 Accuracy for fruit classification	46

LIST OF TABLES

Name of the Table	Page No
Table-4.1 Testing for plant disease detection	37
Table-4.2 Testing for fruit classification	38
Table-5.1 Qualitative analysis for plant disease detection	41
Table-5.2 Qualitative analysis for fruit classification	45

ABSTRACT

In agriculture, early detection and prediction of plant diseases play a crucial role in ensuring crop health and maximizing yield. Traditional methods of disease identification often rely on manual inspection, which can be time-consuming and labor-intensive. In this study, we propose a novel approach utilizing the Faster Region-based Convolutional Neural Network (Faster R-CNN) for automated plant disease detection and fruit prediction. The Faster R-CNN model is trained on a dataset consisting of images of healthy and diseased plants, as well as various types of fruits at different stages of ripeness. By leveraging the power of deep learning and object detection techniques, our model can accurately identify the presence of diseases in plants and predict the ripeness of fruits from input images. Experimental results demonstrate the effectiveness of the proposed method in accurately detecting plant diseases and predicting fruit ripeness, thus offering a promising solution for precision agriculture and enhancing crop management practices.

CHAPTER 1

INTRODUCTION

1.1 Introduction to the project

1.1.1 Plant Disease Detection Overview:

The Plant Disease Detection module is designed to assist farmers and agricultural experts in quickly identifying diseases affecting plant leaves [2]. By uploading images of plant leaves, users can leverage advanced machine learning techniques to detect and classify various types of diseases, enabling early intervention and targeted management strategies.

The primary purpose of this module is to provide a user-friendly tool for automated plant disease diagnosis, aiding farmers in timely disease detection and management. By accurately identifying diseases in plant leaves, the module helps minimize crop losses, optimize resource allocation, and promote sustainable agricultural practices.

Features:

1. Upload plant leaf images for disease detection: Users can upload images of plant leaves affected by diseases directly through the web interface.
2. Detection of various plant diseases: The module utilizes a trained Fast Region-based Convolutional Neural Network (Fast R-CNN) to detect and classify common plant diseases, such as leaf rust, powdery mildew, and blight.
3. Display of disease classification results: Once the image is uploaded, the module provides users with detailed information about the detected disease, including its name.

Usage:

1. Navigate to the Plant Disease Detection module on the web platform.
2. Click on the "Upload Image" button to select the plant leaf image from your device.
3. Wait for the module to process the uploaded image.
4. Once processing is complete, the module will display the results, including the detected disease.
5. Users can take appropriate actions based on the detected disease, such as implementing disease management strategies or seeking further assistance from agricultural experts.

1.1.2 Fruit Classification Overview:

The Fruit Classification module offers a convenient solution for accurately identifying and classifying different types of fruits based on their visual appearance [3]. By uploading images of fruits, users can leverage deep learning algorithms to classify fruits into predefined categories, facilitating quality assessment and sorting processes in agricultural operations.

The main purpose of this module is to provide a reliable tool for fruit classification, supporting farmers, distributors, and food processors in streamlining fruit sorting and grading tasks. By automating the classification process, the module helps improve efficiency, reduce labor costs, and ensure consistency in fruit quality assessment.

Features:

1. Upload fruit images for classification: Users can upload images of fruits captured from various angles and perspectives directly through the web interface.
2. Classification of various fruit types: The module employs a trained convolutional neural network (CNN) to classify fruits into predefined categories, such as apple, banana, orange, and mango, among others.
3. Display of classification results: Upon processing the uploaded image, the module provides users with the predicted fruit category, along with a confidence score indicating the model's certainty.

Usage:

1. Access the Fruit Classification module on the web platform.
2. Use the "Upload Image" button to select the fruit image you want to classify from your device.
3. Wait for the module to analyze the uploaded image.
4. Once analysis is complete, the module will display the predicted fruit category and the associated confidence score.

1.2 Area of the Project

Over the past two decades Machine Learning has become one of the mainstays of information technology and with that, a rather central, albeit usually hidden, part of our life[1]. With the ever-increasing amounts of data becoming available there is good reason to believe that smart data analysis will become even more pervasive as a necessary ingredient for technological progress.

1.3 Problem Definition

- Develop a machine learning model that can accurately detect and classify plant diseases based on images of leaves.
- The model should be able to identify common diseases affecting crops and provide timely recommendations for treatment.
- Additionally, the model can also be extended to classify different types of fruits based on their visual characteristics, aiding in automated fruit sorting and quality control processes.
- This would help farmers detect diseases early, prevent crop losses, and ensure the delivery of high-quality produce to consumers.

1.4 Goal

The primary goal of integrating machine learning techniques for plant disease detection and fruit classification is to develop an efficient system that can assist farmers

- Early Disease Detection
- Crop Monitoring and Management
- Improved Yield and Quality
- Market Efficiency
- Precision Agriculture

1.5 Tools Required

Hardware Requirements:

Processor	:	2 GPUs P100
Monitor	:	LCD/LED
Mouse	:	Optical Mouse
Operating System	:	Windows11

Software Requirements:

Backend	:	Python, Flask
Frontend	:	HTML, CSS, JavaScript
Technology	:	AI/ML

Feasibility Study:

Technical Feasibility:

Algorithm Performance: The feasibility study evaluates the performance of the Fast R-CNN and CNN algorithms in accurately detecting plant diseases and classifying fruits based on images. This includes assessing the models' accuracy, speed, and robustness across different environmental conditions and types of plant diseases/fruits.

Data Requirements: It examines the availability and quality of datasets for training the machine learning models. This involves determining whether sufficient annotated images of plant diseases and fruits are accessible to train and validate the models effectively.

Technological Infrastructure: The study considers the technological infrastructure required to deploy the modules, including hardware resources (e.g., servers, GPUs) and software frameworks (e.g., TensorFlow, Flask). It assesses whether the existing infrastructure can support the computational demands of the project or if additional resources are needed.

Economic Feasibility:

Cost Analysis: The feasibility study conducts a cost-benefit analysis to determine the financial feasibility of the project. This involves estimating the expenses associated with data acquisition, model development, infrastructure setup, and ongoing maintenance. It also considers potential revenue streams or cost savings resulting from the implementation of the modules, such as increased crop yields or labor savings in fruit sorting.

Return on Investment (ROI): It calculates the projected ROI of the project over a specified timeperiod, taking into account the expected benefits and costs. This helps stakeholders assess the financial viability and potential profitability of investing in the project.

Operational Feasibility:

User Acceptance: The study evaluates the acceptability of the modules among end users, including farmers, agricultural experts, and other stakeholders. This involves gathering feedback through user testing and surveys to assess the ease of use, usefulness, and satisfaction with the modules' functionality.

Integration with Existing Systems: It examines the compatibility of the modules with existing agricultural systems and workflows. This includes assessing whether the modules can seamlessly integrate with farm management software, IoT devices, or other tools used in agricultural operations.

Legal and Ethical Feasibility:

Data Privacy and Security: The feasibility study considers legal and regulatory requirements related to data privacy and security. This includes ensuring compliance with regulations such as GDPR (General Data Protection Regulation) and implementing measures to protect user data and maintain confidentiality.

Ethical Considerations: It examines ethical implications associated with the use of machine learning models in agriculture, such as potential biases in the models' predictions, data transparency, and accountability. This involves addressing concerns related to fairness, equity, and social impact.

Based on the findings of the feasibility study, stakeholders can make informed decisions regarding the implementation of the project. If the study demonstrates that the project is technically feasible, economically viable, operationally practical, and compliant with legal and ethical requirements, it may proceed to the development and deployment phases. However, if significant challenges or risks are identified, adjustments may be necessary to ensure the project's success and sustainability.

1.6 Organization of the Project

This report fully describes the project under taken. However, to control the length of the report, the reader is an occasion referred to a bibliographical reference if particular details are required in an area. The report is split into six main sections:

Chapter-1, Introduction – It gives an overview of the project i.e., problem definition, goals to overcome that problem and the prerequisites required to develop the model. This system is designed and developed by the usage of the different machine learning models available.

Chapter-2, Background – This section mainly deals with the basics required for the python programming and different machine learning models. It also deals with the different packages required for modeling the system. Further, possible extensions to the basic design requirements are proposed.

Chapter-3, Proposed Method – In this, the complete process for the successful execution of the project is explained. It gives the brief explanation of Faster R-CNN model.

Chapter-4, Software Testing - Evaluation of the success of the implementation in meeting the needs discovered in the background research is given in quantitative and qualitative terms. The system is compared to others whether it meets its performance characteristics.

Chapter-5, Results – Various performance qualitative and quantitative shown. It shows how accurately the Faster R-CNN works for analyzing the dataset does.

Chapter-6, Conclusions - Analysis of the successes and failures of the project and discussion of the advances made.

CHAPTER 2

BACKGROUND

2.1 Overview of Technologies used

2.1.1 Convolutional Neural Networks (CNNs)

- Introduction to CNNs and their role in deep learning-based image processing tasks.
- Explanation of CNN architecture [4], including convolutional layers, pooling layers, and fully connected layers.
- Advantages of CNNs in feature extraction and hierarchical representation learning.
- Implementation considerations for training and fine-tuning CNNs for specific tasks such as plant disease detection and fruit classification.

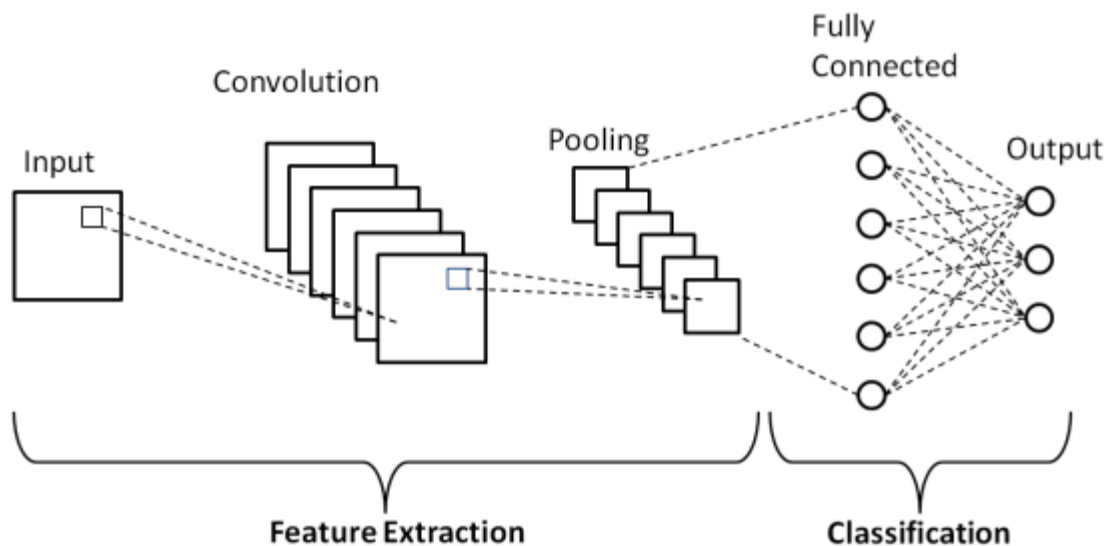


Figure 2.1. Schematic Diagram for CNN

- CNN consists of convolutional layer, pooling layer and fully connected layer as shown in Figure 2.1.

Convolutional Layer

This layer is the first layer that is used to extract the various features from the input images.

Pooling Layer

In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs.

Fully Connected Layer

The Fully Connected (FC) layer consists of the weights and biases along with the neurons and is used to connect the neurons between two different layers.

2.1.2 Region-based Convolutional Neural Networks (R-CNNs)

- Introduction to R-CNNs and their evolution from traditional CNNs.
- Explanation of the R-CNN architecture, including region proposal generation and region-wise classification[6].
- Overview of the challenges addressed by R-CNNs, such as the need for accurate object localization and efficient computation.
- Comparison of R-CNN variants, including Fast R-CNN, Faster R-CNN, and Mask R-CNN.

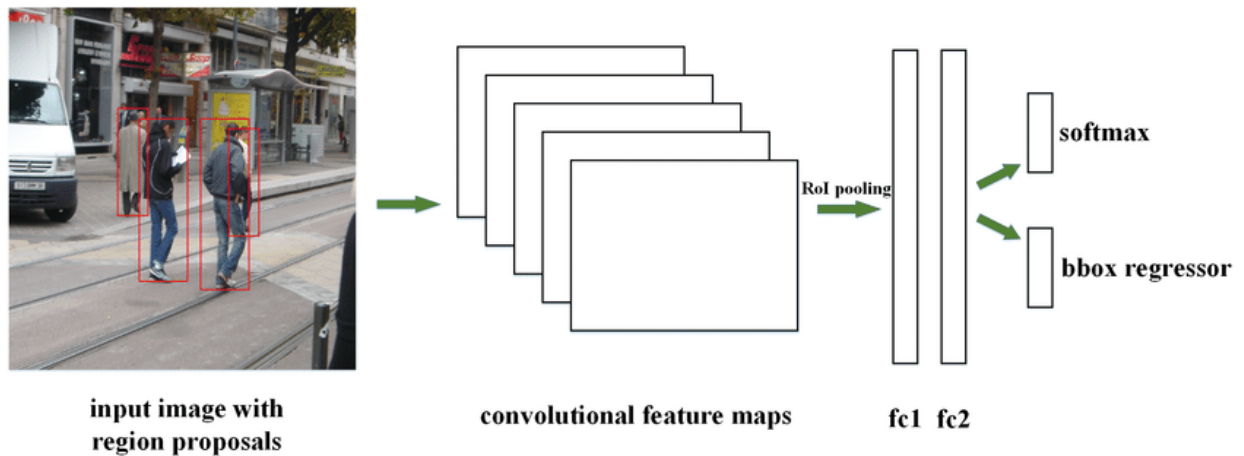


Figure 2.2. Schematic Diagram for R-CNN

- R-CNN consists of region Proposal layer, convolutional layer, polling layer ass shown in Figure 2.2.

Region Proposals

Region proposals are simply the smaller regions of the image that possibly contains the objects we are searching for in the input image. To reduce the region proposals in the R-CNN uses a greedy algorithm called selective search.

Convolutional Layer

This layer is the first layer that is used to extract the various features from the input images.

Pooling Layer

In most cases, a Convolutional Layer is followed by a Pooling Layer. The primary aim of this layer is to decrease the size of the convolved feature map to reduce the computational costs.

Bounding Box Regressor

In order to precisely locate the bounding box in the image., we used a scale-invariant linear regression model called bounding box regressor.

Softmax

Softmax is typically used in the last layer of a neural network to predict the class of an input.

2.1.3 Fast Region-based Convolutional Neural Network (Fast R-CNN)

- Detailed explanation of Fast R-CNN architecture and workflow, emphasizing its improvements over traditional R-CNN.
- Description of the Region Proposal Network (RPN) [7] and its role in generating region proposals.
- Explanation of the RoI pooling layer for feature extraction from region proposals.
- Advantages of Fast R-CNN, including faster inference speed and improved accuracy.
- In Fast R-CNN the image is given to only one CNN, then the CNN is given to ROI.
- The Region of Interest (ROI) [9] can estimate the location of every object. So, CNN is given to ROI.
- The time taken for Fast R-CNN is 2 sec and speed is 25x.
- Figure 2.3, shows the architecture of Fast R-CNN.

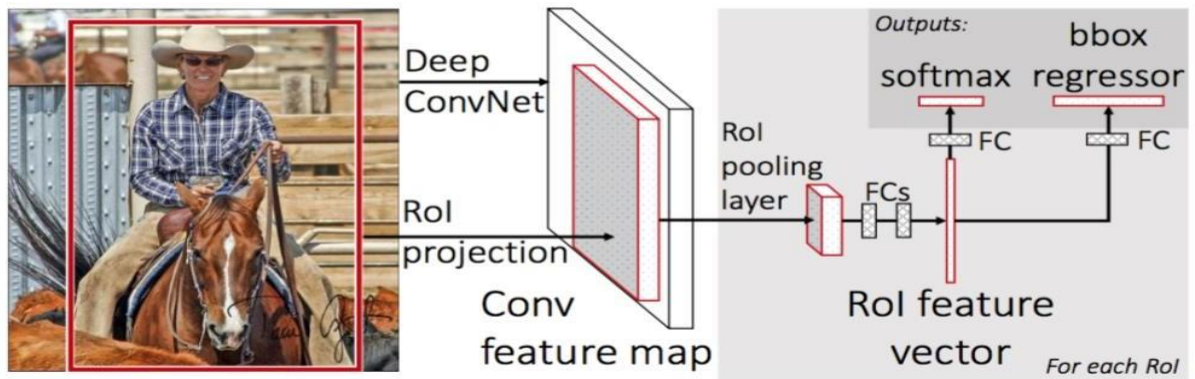


Figure 2.3. Fast R-CNN Architecture

From Figure 2.3. First, we generate the region proposal from a selective search algorithm. This selective search algorithm generates up to approximately 2000 region proposals. These region proposals (RoI projections) combine with input images passed into a CNN network. This CNN network generates the convolution feature map as output. Then for each object proposal, a Region of Interest (RoI) pooling layer extracts the feature vector of fixed length for each feature map. Every feature vector is then passed into twin layers of softmax classifier and Boundary Box(Bbox) regression for classification of region proposal and improve the position of the bounding box of that object.

2.1.4 Technologies for Model Training and Deployment

- TensorFlow: Introduction to TensorFlow and its role as a popular deep learning framework for model development and training [10].
- PyTorch: Overview of PyTorch and its advantages, including dynamic computation graphs and ease of use.
- Training Considerations: Discussion on data preprocessing, model architecture selection, hyperparameter tuning, and training strategies for Faster R-CNN models.
- Deployment Considerations: Explanation of model deployment options, including cloud-based solutions, edge computing, and containerization.

2.1.5 Web Development Technologies for User Interface

- Flask: Introduction to Flask as a lightweight and versatile web framework for Python.
- HTML, CSS, and JavaScript: Overview of front-end technologies for building user interfaces.
- Integration with Deep Learning Models: Explanation of how Flask can be used to serve deep learning models for inference through RESTful APIs.

2.1.6 Image Processing Libraries and Tools

- OpenCV: Introduction to OpenCV as a popular library for computer vision tasks, including image pre-processing and manipulation.
- PIL (Python Imaging Library): Overview of PIL and its role in image loading, resizing, and augmentation.
- TensorFlow Object Detection API: Explanation of TensorFlow Object Detection API for streamlined development and deployment of object detection models.

2.1.7 Data Management Technologies

- PostgreSQL: Introduction to PostgreSQL as a relational database management system (RDBMS) for data storage and retrieval.
- MongoDB: Overview of MongoDB as a NoSQL database for handling unstructured or semi-structured data.
- Data Annotation Tools: Discussion on tools and platforms for annotating image datasets for training deep learning models.

2.1.8 Cloud Computing and Scalability

- Amazon Web Services (AWS): Introduction to AWS cloud computing platform and its services relevant to machine learning, such as Amazon SageMaker and EC2[11].
- Google Cloud Platform (GCP): Overview of GCP services for machine learning, including Google Cloud AI Platform and Compute Engine.
- Scalability Considerations: Discussion on scaling deep learning models and web applications to handle increased workloads and user traffic.

2.1.9 Ethical and Legal Considerations

- **Data Privacy:** Explanation of data privacy regulations such as GDPR and considerations for handling sensitive user data.
- **Bias and Fairness:** Discussion on mitigating bias in machine learning models and ensuring fairness in algorithmic decision-making.
- **Intellectual Property:** Overview of intellectual property rights related to code, models, and datasets, and considerations for licensing and copyright.

2.2 Python

Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. Here's an explanation covering various aspects of Python:

2.2.1 General Overview:

Python was created by Guido van Rossum and first released in 1991. It is open-source and has since become one of the most popular programming languages worldwide.

Python's design philosophy emphasizes readability and simplicity, making it accessible to beginners while also powerful enough for complex applications.

Python is widely used across diverse domains, including web development, data analysis, artificial intelligence, scientific computing, automation, and more.

2.2.2 Syntax and Structure:

Python uses a clean and straightforward syntax with minimal punctuation, relying on indentation to indicate blocks of code.

Variables in Python are dynamically typed, meaning their types are inferred at runtime, allowing for flexible and concise code.

Python supports object-oriented, imperative, functional, and procedural programming paradigms, providing developers with flexibility in designing and structuring their code.

2.2.3 Features:

Readability: Python's syntax is designed to be readable and intuitive, making it easy to write and understand code.

Extensive Standard Library: Python comes with a comprehensive standard library that provides modules and functions for a wide range of tasks, from file I/O to networking to mathematical operations.

Interpreted and Interactive: Python code is interpreted line by line, allowing for interactive development through tools like Python's interactive shell (REPL).

Portability: Python is platform-independent, meaning code written in Python can run on various operating systems without modification.

Community and Ecosystem: Python has a vibrant and active community that contributes libraries, frameworks, and resources to the Python ecosystem, fostering innovation and collaboration.

2.2.4 Applications:

Web Development: Python is used for server-side web development with frameworks like Django and Flask, enabling the creation of dynamic and scalable web applications.

Data Science and Machine Learning: Python is widely used in data analysis, machine learning, and artificial intelligence applications, with popular libraries like NumPy, Pandas, TensorFlow, and PyTorch.

Scripting and Automation: Python is often used for scripting and automation tasks, such as system administration, batch processing, and task automation.

Scientific Computing: Python is a popular choice for scientific computing and numerical simulations due to its extensive libraries and ease of use in prototyping and experimentation.

2.2.5 Tools and Ecosystem:

Package Management: Python's package management system, pip, allows developers to easily install and manage third-party libraries and dependencies.

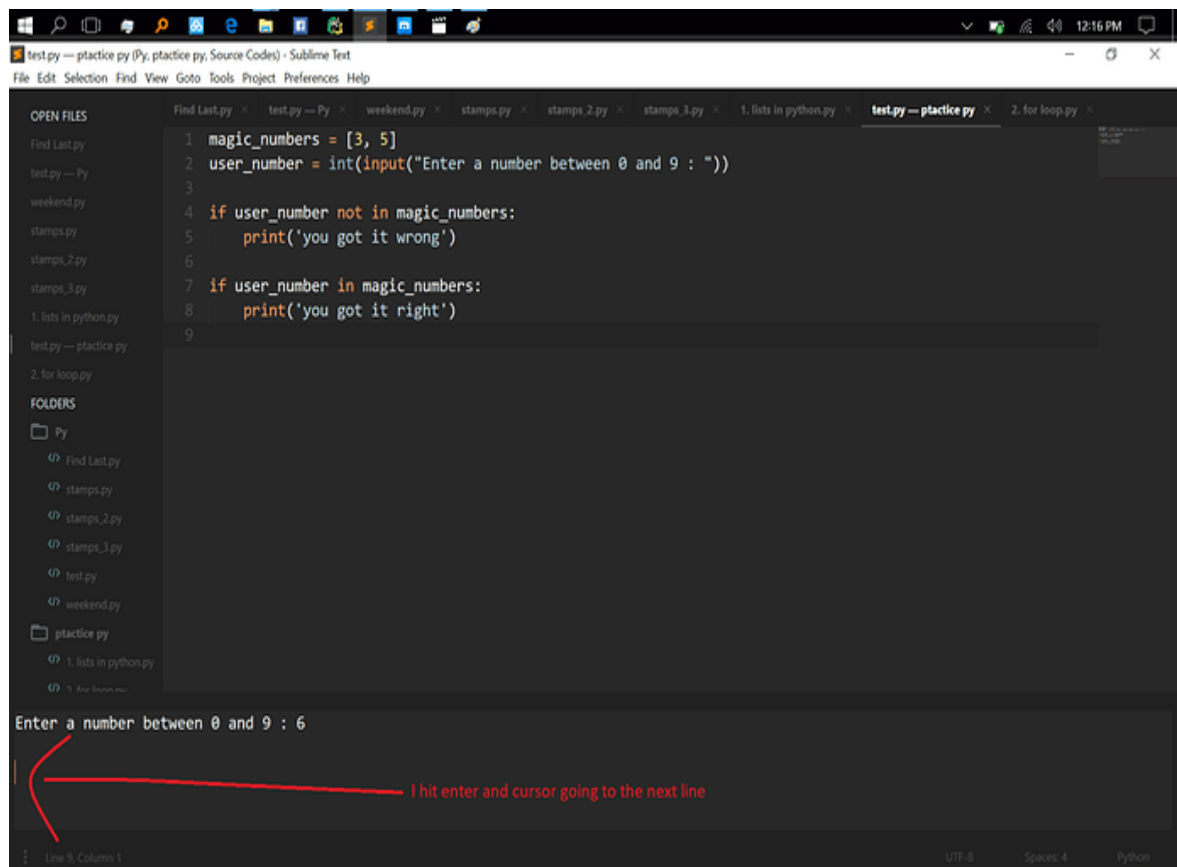
Integrated Development Environments (IDEs): Python supports a variety of IDEs, including PyCharm, VS Code, and Jupyter Notebook, which provide features like code completion, debugging, and project management.

Testing and Debugging: Python offers built-in testing frameworks like unittest and pytest for writing and running tests, as well as debugging tools like pdb and integrated debuggers in IDEs.

2.2.6 Recent Developments:

Python 3: Python 3 is the latest major version of Python, introducing improvements and new features over Python 2. Python 2 reached its end of life in January 2020, and developers are encouraged to migrate to Python 3 for ongoing support and compatibility.

Performance Optimization: Efforts are ongoing to improve Python's performance, with initiatives like PEP 554 (Multiple Interpreters in the Stdlib) and projects like PyPy and Cython offering alternative Python interpreters and compilers for performance optimization.



```

1 magic_numbers = [3, 5]
2 user_number = int(input("Enter a number between 0 and 9 : "))
3
4 if user_number not in magic_numbers:
5     print('you got it wrong')
6
7 if user_number in magic_numbers:
8     print('you got it right')
9

```

Enter a number between 0 and 9 : 6

I hit enter and cursor going to the next line

Figure 2.4. Python 3

- We used Python 3 to write code as shown in the figure 2.4.

CHAPTER 3

PROPOSED METHOD

Contribution of the Project

The main contribution of using Faster R-CNN (Region-based Convolutional Neural Network) for plant disease detection and fruit classification lies in its effectiveness and efficiency in automating these processes. Faster R-CNN is a deep learning-based object detection algorithm that combines the capabilities of both region proposal networks and convolutional neural networks to accurately identify objects within images.

In the context of plant disease detection, Faster R-CNN can analyze images of plants and accurately identify regions of interest (e.g., leaves) that exhibit symptoms of diseases. By automating this process, agricultural experts can quickly assess the health of plants in large fields, allowing for early detection and intervention to prevent the spread of diseases and minimize crop losses.

Similarly, in fruit classification, Faster R-CNN can be trained to recognize different types of fruits based on their visual characteristics such as shape, color, and texture. This can be particularly useful in fruit sorting and grading processes, where fruits need to be categorized based on various quality criteria. By automating fruit classification, Faster R-CNN can improve the efficiency of sorting processes in agricultural settings, reducing labor costs and ensuring consistent quality standards.

Overall, the main contribution of using Faster R-CNN for plant disease detection and fruit classification is its ability to automate these tasks with a high degree of accuracy and efficiency.

3.1 Proposed System:

The proposed system for plant disease detection and fruit classification leverages advanced machine learning techniques, specifically Faster R-CNN for disease detection and CNN for fruit classification, to automate and enhance these processes. The system consists of two modules:

1. Plant Disease Detection Module:

- Users can upload images of plant leaves affected by diseases through a user-friendly web interface.
- The system utilizes Faster R-CNN, trained on a dataset of annotated plant images, to accurately detect and classify various plant diseases in the uploaded images.
- Results are displayed to users, providing information about the detected disease(s).
- This module enables farmers and agricultural experts to quickly identify and manage plant diseases, leading to improved crop health and productivity.

2. Fruit Classification Module:

- Users can upload images of fruits for classification purposes via the web interface.
- The system employs a convolutional neural network (CNN), trained on a dataset of annotated fruit images, to classify fruits into predefined categories based on their visual appearance.
- Classification results, including the predicted fruit category and confidence score, are presented to users.
- This module facilitates automated fruit sorting and grading, enabling farmers and food processors to optimize product quality and market competitiveness.

3.2 Key Advantages of the Proposed System:

- **Automation:** The proposed system automates the processes of plant disease detection and fruit classification, reducing the reliance on manual labor and subjective assessments.
- **Accuracy:** By leveraging advanced machine learning algorithms, the system enhances the accuracy and reliability of disease detection and fruit classification, leading to more precise decision-making.
- **Efficiency:** The automated nature of the system improves operational efficiency, enabling faster detection of plant diseases and sorting of fruits, thus saving time and resources.
- **Scalability:** The proposed system can be scaled to accommodate large volumes of data and cater to the needs of diverse agricultural settings, from small-scale farms to commercial production facilities.

Overall, the proposed system offers a comprehensive and efficient solution for plant disease detection and fruit classification, empowering farmers and agricultural stakeholders with valuable insights for improved crop management and productivity.

3.3 Block Diagram of Faster R-CNN

Below is an outline for a document explaining the technologies used in a project involving Faster R-CNN for plant disease detection and fruit classification. The document aims to provide a comprehensive overview of the technologies, including their roles, advantages, and implementation considerations.

The block diagram for Faster R-CNN is as shown in the Figure 3.1

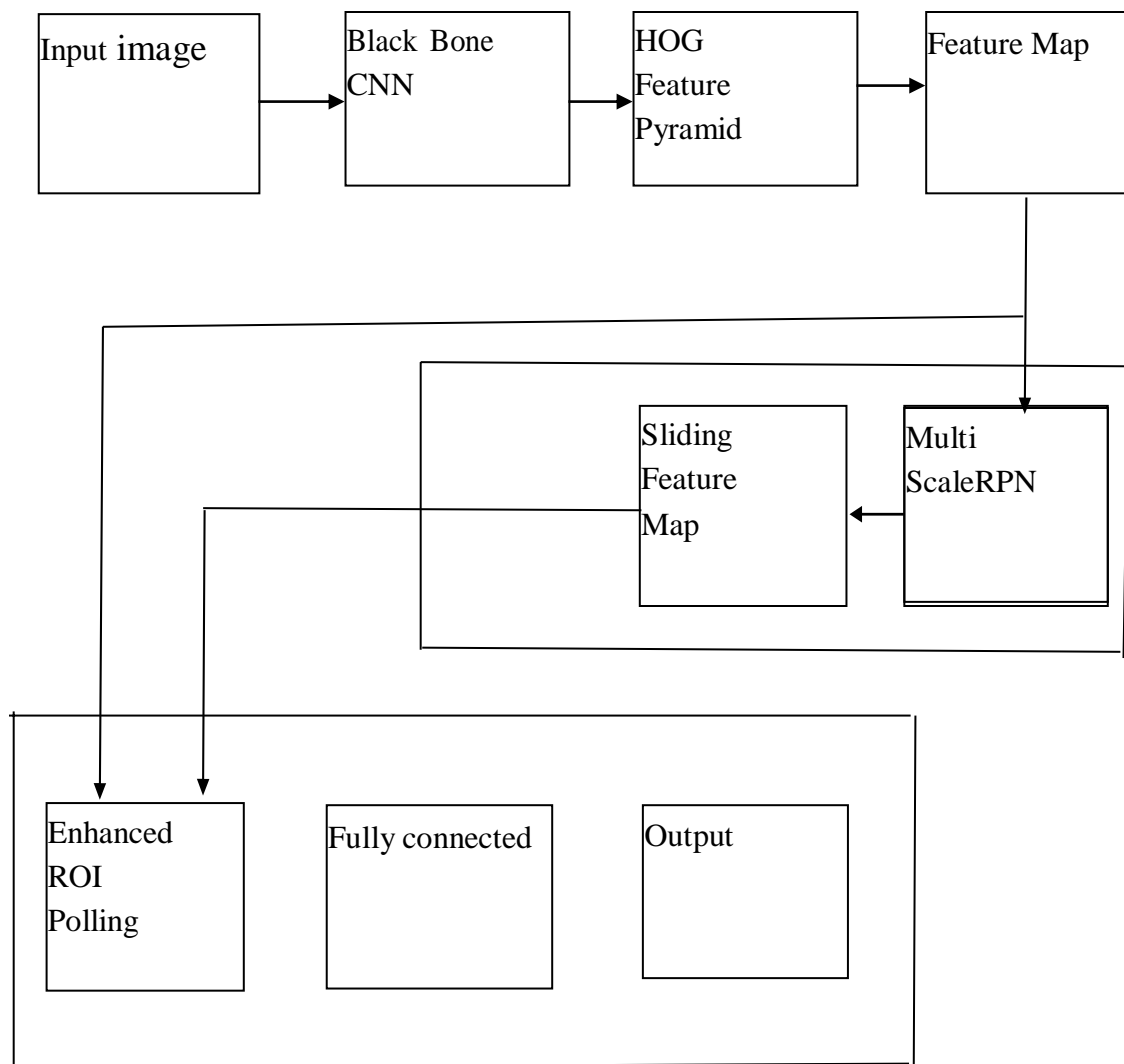


Figure 3.1. Block diagram of Faster R-CNN

3.3.1 Explanation

Faster R-CNN (Region-based Convolutional Neural Network) is a popular deep learning-based object detection framework proposed by Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun in 2015. It represents a significant advancement over earlier object detection techniques by combining deep learning with region proposal methods in a single end-to-end trainable model.

Faster R-CNN architecture consists of two components

1. Region Proposal Network (RPN)
2. Fast R-CNN detector

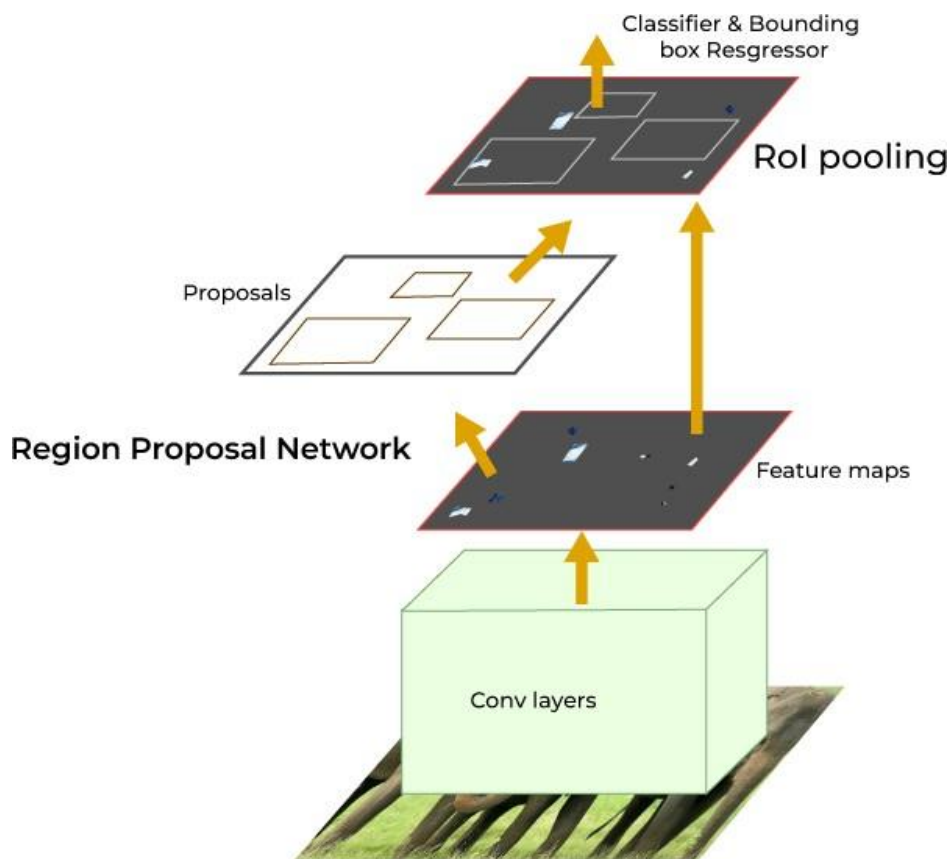


Figure 3.2. Faster R-CNN Architecture

Before discussing the RPN and Fast R-CNN detector, Let's understand the Shared Convolutional Layers that works as the backbone in Faster R-CNN architecture. It is the common CNN layer used for both RPN and Fast R-CNN detector as shown in the figure.

Convolutional Neural Network (CNN) Backbone

The Convolutional Neural Network (CNN) Backbone is the starting layers of Faster R-CNN architecture. The input image is passed through CNN backbone (e.g., ResNet, VGG) to extract feature maps. These feature maps capture different levels of visual information from the image. Which is further used by Region Proposal Network (RPN) and Fast R-CNN detector. Let's understand the role of Convolutional Neural Network (CNN) Backbone in details

- The primary objective of CNN is to extract the relevant features from the input image. It consists of multiple convolutions layers that apply different-different convolutions kernel to extract the feature from the input image.
- These kernels are designed to capture the hierarchical representations of the input image means the initial layers of CNN captures the low-level features likes edges and textures, and while deeper layers captures the high level semantic features like objects parts and shapes.
- Both RPN and Fast R-CNN detector uses the same extracted hierarchical features. This results in a significant reduction in computing time and memory use because the computations carried out by these layers are employed for both tasks.

1. Region Proposal Network (RPN):

- Faster R-CNN introduces a Region Proposal Network (RPN) [12] that operates on the convolutional feature maps of a CNN.
- The RPN generates a set of region proposals, which are candidate bounding boxes that potentially contain objects of interest.
- These proposals are generated by sliding a small network, typically a fully convolutional network (FCN), over the feature map to predict regions likely to contain objects.

2. Region of Interest (RoI) Pooling:

- Once the region proposals are generated, they are fed into a Region of Interest (RoI) pooling layer[13].
- The RoI pooling layer extracts fixed-size feature maps from the convolutional feature maps for each region proposal, regardless of its size or aspect ratio.
- This ensures that the extracted features are spatially aligned and can be fed into subsequent layers for classification and bounding box regression.

3. Classification and Bounding Box Regression:

- The RoI features extracted from the RoI pooling layer are passed through fully connected layers for classification and bounding box regression [14].
- Classification involves predicting the probability of each region proposal containing an object of interest, typically using softmax activation.
- Bounding box regression predicts adjustments to the coordinates of the bounding boxes to refine their positions and sizes, improving localization accuracy.

4. Training:

- Faster R-CNN is trained end-to-end using a multi-task loss function that combines classification loss (e.g., cross-entropy loss) and regression loss (e.g., smooth L1 loss).
- The model is trained on a dataset with annotated bounding boxes for various object classes, such as the PASCAL VOC or COCO datasets.
- During training, the parameters of both the RPN and the subsequent classification and regression layers are optimized jointly using backpropagation and gradient descent.

Overall, Faster R-CNN has become a cornerstone in the field of object detection, providing a powerful and efficient framework for detecting objects in images with high accuracy and speed. Its modular architecture and end-to-end training make it highly versatile and suitable for a wide range of applications in computer vision.

The Faster R-CNN model consists of two main components: the Region Proposal Network (RPN) and the Region of Interest (RoI) pooling layer for classification and bounding box regression.

3.3.2 Region Proposal Network (RPN):

The RPN predicts object bounding boxes (region proposals) and objectness scores for candidate regions in the input image. It generates region proposals by sliding a small network over the convolutional feature map.

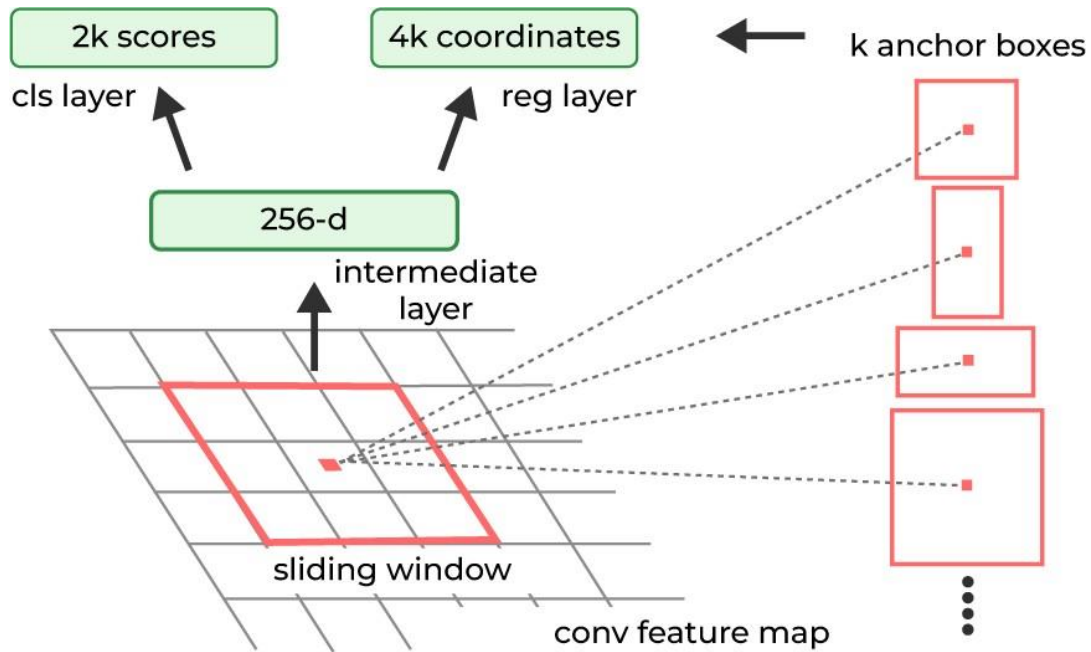


Figure 3.3. Region Proposal Network (RPN)

Anchors boxes: Anchors are used to generate region proposals in the Faster R-CNN model as shown in figure 3.3. It uses a set of predefined anchor boxes with various scales and aspect ratios [15]. These anchorboxes are placed at different positions on the feature maps. An anchor box has two key parameters

- Scale
- aspect ratio

An anchor box is a predefined box of various scales and aspect ratios that serves as the basis for region proposal generation. Let (x_i, y_i) denote the center coordinates of the anchor box, w_i and h_i denote its width and height, and s denote the scale factor.

The anchor box coordinates and dimensions are calculated as follows:

$$X_i = \text{Pixel_Scale} * \text{Scale} * (\text{Col} + 0.5)$$

$$Y_i = \text{Pixel_Scale} * \text{Scale} * (\text{Col} + 0.5)$$

$$W_i = \text{Pixel_Scale} * \text{Scale} * \text{Aspect_ratio_width}$$

$$H_i = \text{Pixel_Scale} * \text{Scale} * \text{Aspect_ratio_height}$$

Sliding Window approach: The RPN operates as a sliding window mechanism over the feature map obtained from the CNN backbone[16]. It uses a small convolutional network (typically a 3×3 convolutional layer) to process the features within the receptive field of the sliding window. This convolutional operation produces scores indicating the likelihood of an object's presence and regression values for adjusting the anchor boxes.

Region Proposal Formulation:

For each anchor box, the RPN predicts two values: the probability of objectness (p) and the bounding box offsets ($\Delta X, \Delta Y, \Delta W, \Delta H$).

$$P_i = \text{sigmoid}(\text{score_logit})$$

$$\Delta X_i, \Delta Y_i, \Delta W_i, \Delta H_i = \text{regression_output}$$

Where P_i is the probability of objectness for the i th anchor box.

3.3.3 Region of Interest (RoI) Pooling Layer:

The RoI pooling layer extracts fixed-size feature maps from the convolutional feature maps for each region proposal generated by the RPN.

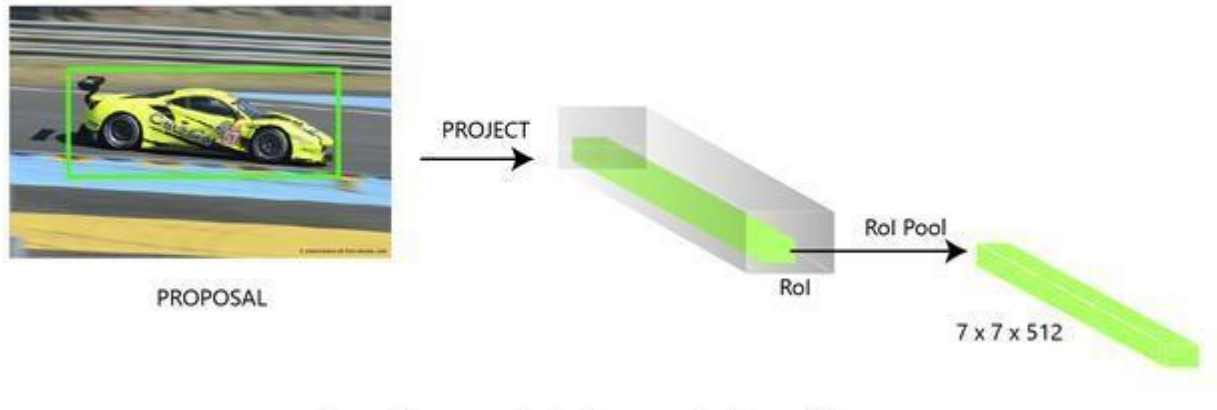


Figure 3.4. Region of Interest (ROI) pooling

RoI Pooling Formula:

For each region proposal, the RoI pooling layer divides the region into $N \times N$ sub-windows and performs max pooling over each sub-window to obtain a fixed-size feature map.

$$X_{\text{pooled}} = 1/N \sum_i \max(0, X_i - X_{\min}) \quad Y_{\text{pooled}} = 1/N \sum_i \max(0, \hat{Y}_i - Y_{\min})$$

$$\text{Pooled_feature_map}[X_{\text{pooled}}, Y_{\text{pooled}}, K] =$$

$$\max(\text{Pooled_feature_map}[X_{\text{pooled}}, Y_{\text{pooled}}, K], \text{feature_map}[X_i, Y_i, K])$$

where X_{\min} , Y_{\min} denote the top-left corner coordinates of the region proposal, X_i , \hat{Y}_i denote the coordinates of the sub-window and $\text{pooled_feature_map}$ denotes the output feature map.

These formulas describe the key components of the Faster R-CNN model for object detection. Implementing these formulas in code requires understanding how to formulate the anchor boxes, compute objectness scores, predict bounding box offsets, and perform RoI pooling efficiently.

Feature Extraction: The RoI-pooled feature maps are fed into the CNN backbone (the same one used in the RPN for feature extraction)[17][23] to extract meaningful features that capture object-specific information. It draws hierarchical features from region proposals. These features retain spatial information while abstracting away low-level details, allowing the network to understand the proposed regions' content.

Fully Connected Layers: The RoI-pooled and feature-extracted regions then pass through a series of fully connected layers[18]. These layers are responsible for object classification and bounding box regression tasks.

Object Classification: The network predicts class probabilities for each region proposal, indicating the possibility that the proposal contains an object of a specific class. The classification is carried out by combining the features retrieved from the region proposal with the shared weights of the CNN backbone.

Bounding Box Regression: In addition to class probabilities, the network predicts bounding box adjustments for each region proposal[19]. These adjustments refine the position and size of the bounding box of the region proposal, making it more accurate and aligned with the actual object boundaries.

The first layer is a softmax layer of $N+1$ output parameters (N is the number of class labels and background) that predicts the objects in the region proposal. The second layer is a bounding box regression layer that has $4 * N$ output parameters. This layer regresses the bounding box location of the object in the image.

3.3.4 Advantages:

Faster R-CNN offers several advantages over earlier object detection methods:

- It achieves state-of-the-art accuracy on benchmark datasets while being computationally efficient.
- By sharing convolutional features between the RPN and the subsequent detection layers, it reduces redundant computation and improves speed.
- The end-to-end trainable architecture simplifies the training process and enables faster convergence.

3.3.5 Applications:

Faster R-CNN has been widely used in various applications, including but not limited to:

- Object detection in images for tasks such as pedestrian detection, vehicle detection, and aerial imaging.
- Instance segmentation, where each object instance is segmented and classified individually.
- Semantic segmentation, where each pixel in an image is classified into different classes.

CHAPTER 4

SOFTWARE TESTING

4.1 Dataset

4.1.1 Data set for plant disease detection

Due to simplicity in design we had gathered a syntactic data could have features like color, shape, and texture like real plant leaves[2]. Based on Figure 4.1, it can be observed that the syntactic data are displaying the desired features.

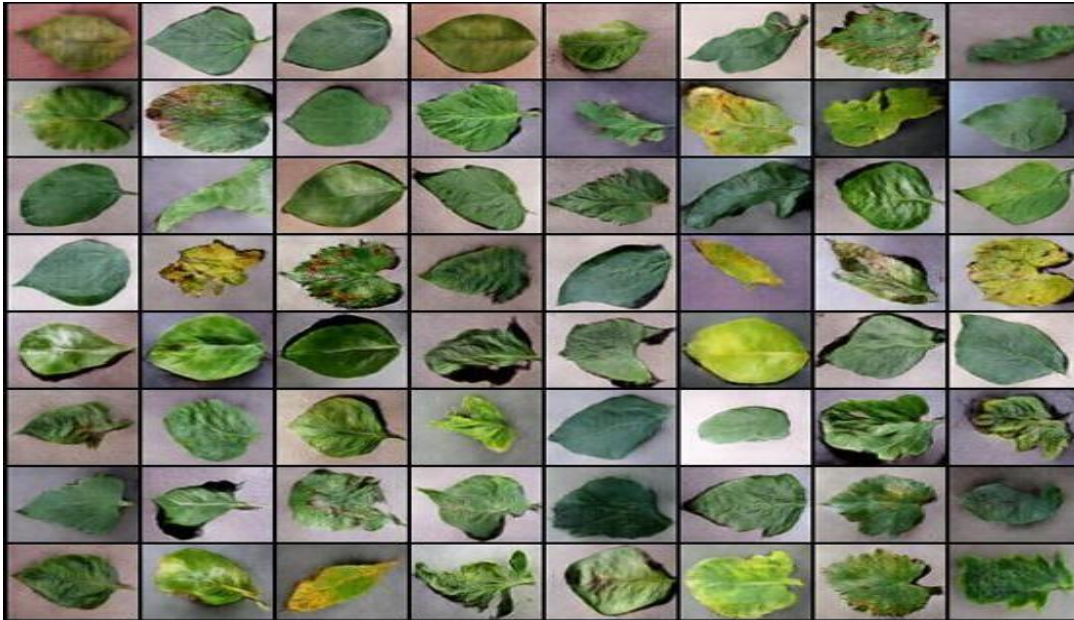


Figure 4.1. Dataset for plant disease detection

4.1.2 Dataset for fruit classification

Figure 4.2, indicates the types of fruits gathered from the internet for training[3].

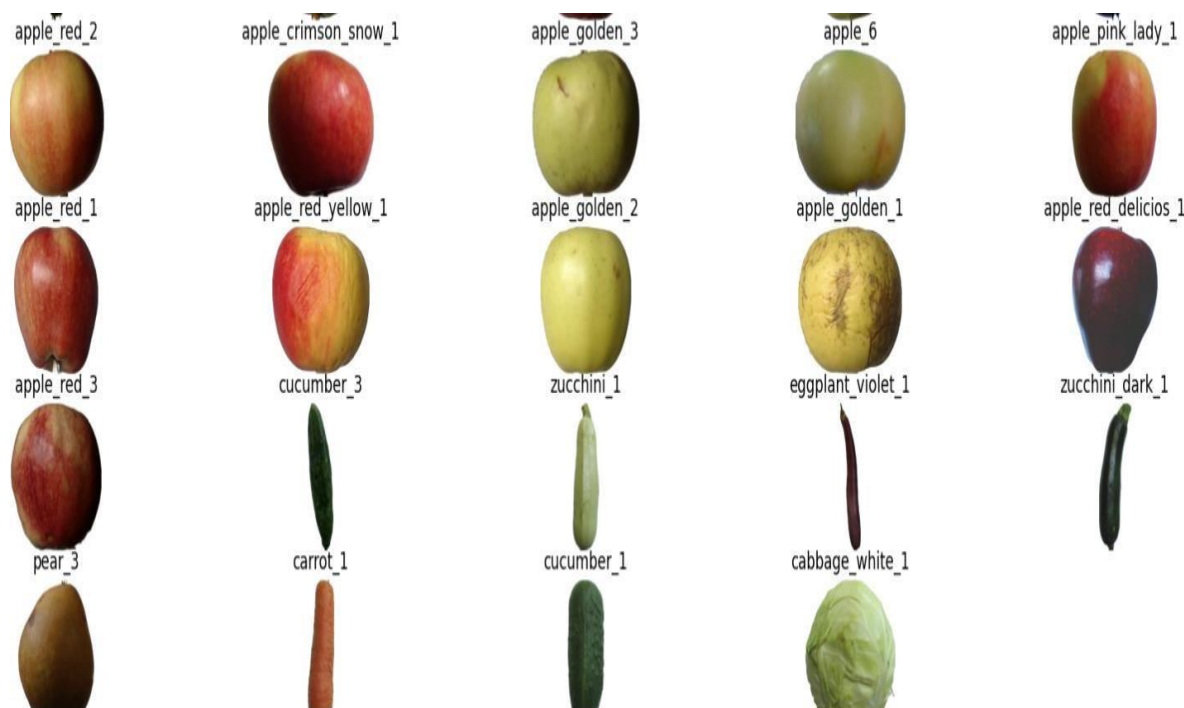


Figure 4.2. Dataset for fruit classification

All the images were collected in JPG format, with a fixed resolution of 1024×768. The images in this dataset are in RGB, where each color channel contains 8-bits per pixel. The images were captured at different dates and times. Figure 5 shows sample images from this dataset.

4.2 TESTING

In the context of a project involving Faster R-CNN for plant disease detection and fruit classification, various types of testing are crucial to ensure the accuracy, robustness, and reliability of the system. Here are the types of testing relevant to this project:

1. Unit Testing:

Test Coverage: Ensure that unit tests cover a significant portion of the codebase, including edge cases and error handling scenarios.

Mocking: Use mocking frameworks to simulate external dependencies and isolate the unit under test, allowing for controlled testing environments.

Continuous Integration: Integrate unit tests into a continuous integration pipeline to automate testing and catch regressions early in the development process.

2. Integration Testing:

API Testing: Verify the correctness of API endpoints and data exchange protocols between frontend and backend components.

Data Integrity: Test data integrity during integration to ensure that data passed between components remains consistent and accurate.

Dependency Management: Validate compatibility and versioning of dependencies to prevent conflicts and ensure smooth integration.

3. System Testing:

EndtoEnd Scenarios: Create comprehensive test cases that mimic realworld usage scenarios, covering all major functionalities and user interactions.

CrossBrowser Testing: Test the web interface on a variety of browsers and versions to ensure compatibility and consistent behavior across platforms.

Accessibility Testing: Verify that the system is accessible to users with disabilities and complies with accessibility standards (e.g., WCAG).

4. Regression Testing:

Test Automation: Implement automated regression tests to efficiently rerun test cases after code changes and detect potential regressions.

Regression Suites: Maintain a suite of regression tests that cover critical functionalities and frequently executed scenarios to ensure stability.

Version Control Integration: Integrate regression testing with version control systems to automatically trigger tests upon code commits and merges.

5. Performance Testing:

Load Testing: Simulate heavy user traffic and workload to assess system performance under peak usage conditions and identify performance bottlenecks.

Scalability Testing: Evaluate the system's ability to scale horizontally and vertically to handle increased loads and resource demands.

Resource Monitoring: Monitor system resource utilization (CPU, memory, disk I/O) during performance tests to identify resource constraints and optimize performance.

6. Usability Testing:

User Feedback Collection: Gather feedback from real users through surveys, interviews, or usability testing sessions to identify usability issues and improvement opportunities.

Heatmap Analysis: Use heatmap tools to visualize user interactions and identify areas of interest, confusion, or frustration within the user interface.

Iterative Design: Incorporate user feedback into the design process to iteratively improve the usability and user experience of the system.

7. Acceptance Testing:

Acceptance Criteria Definition: Establish clear and measurable acceptance criteria based on stakeholder requirements to guide acceptance testing.

User Acceptance Testing (UAT): Involve stakeholders in UAT to validate that the system meets their expectations and fulfills business objectives.

Bug Triaging: Prioritize and address issues identified during acceptance testing based on severity, impact, and stakeholder feedback.

8. Security Testing:

Vulnerability Scanning: Use automated security scanning tools to identify common vulnerabilities such as SQL injection, crosssite scripting (XSS), and authentication flaws.

Threat Modeling: Conduct threat modeling exercises to identify potential security threats, assess their likelihood and impact, and prioritize mitigation efforts.

Secure Coding Practices: Enforce secure coding practices (e.g., input validation, secure authentication) and adhere to security best practices to minimize security risks.

9. Compatibility Testing:

Device Fragmentation: Consider the wide range of devices (desktops, laptops, tablets, smartphones) and operating systems (Windows, macOS, Linux, iOS, Android) in compatibility testing.

Browser Compatibility Tools: Utilize browser compatibility testing tools and services to automate testing across multiple browsers and versions.

Responsive Design: Ensure that the web interface is responsive and adapts gracefully to different screen sizes and resolutions to maintain a consistent user experience.

10. Data Quality Testing:

Data Validation: Implement data validation checks to ensure the quality and integrity of input data, preventing issues such as data corruption or invalid inputs.

Data Profiling: Use data profiling tools to analyze and understand the characteristics of the dataset, including data distributions, outliers, and anomalies.

Anomaly Detection: Employ anomaly detection techniques to identify unusual patterns or outliers in the data that may indicate data quality issues or anomalies.

Each type of testing plays a crucial role in ensuring the quality, reliability, and performance of the system for plant disease detection and fruit classification.

4.2.1 Testing for Plant Disease Detection

Table 4.1, shows the testing results for Plant Disease Detection

Table 4.1. Testing results for Plant Disease Detection

Epoch No	Accuracy	Losses
1	0.1047	3.2731
5	0.3632	2.1803
8	0.5872	1.3447
10	0.6484	1.1386
12	0.6899	0.9940
49	0.8653	0.4319
50	0.8661	0.4314

4.2.2 Testing for Fruit Classification

Table 4.2. shows the testing results for Fruit Classification

Table 4.2. Testing Results for Fruit Classification

Epoch No	Accuracy	Losses
1	0.1187	2.8579
5	0.6648	0.9442
8	0.7819	0.6236
10	0.8366	0.4559
12	0.8769	0.3677
49	0.9678	0.1056
50	0.9756	0.0739

CHAPTER 5

RESULTS

5.1 Plant Disease Detection results

5.1.1 Qualitative results for plant disease detection

This architecture is designed for generating 64×64 images, while training for higher resolutions is highly unstable as one network becomes stronger than the other, which prevents the learning process. In order to use this syntactic data in the training phase, higher resolution images must be generated as most of the state-of-the-art convolutional neural networks are designed for input image sizes around 256×256 .

In order to explore the learned features, the guided backpropagation technique was used. This method uses a simple backward pass of the activation of a single neuron after a forward pass through the network to visualize the part of an image that mostly activates a certain neuron.

Visualization of the generated image is displayed in Figure 5.1.

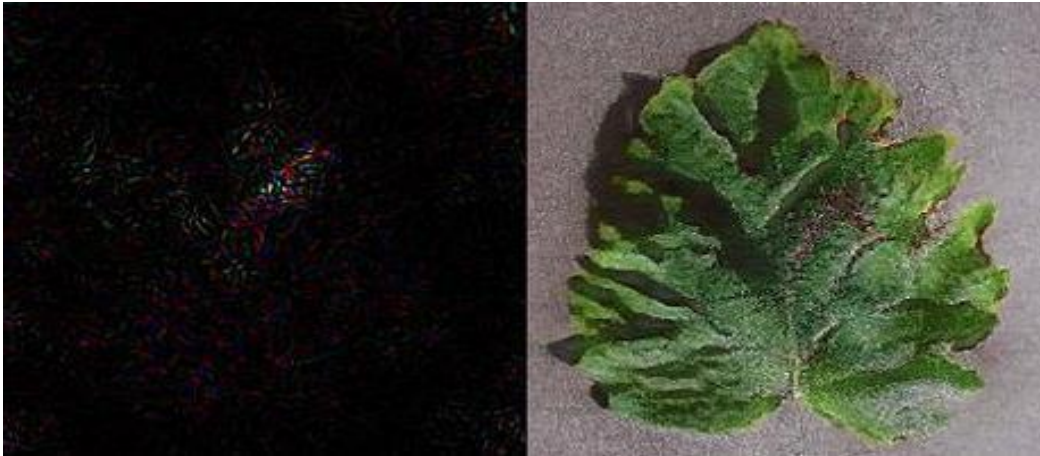


Figure 5.1. Feature visualization

Based on Figure 5.1, it can be observed that Faster R-CNN has learnt significant features representing the diseased areas.

Plant disease detection models could have practical usage where they could be part of a decision support system applied on fields by experts or hobbyists. In order to do this, developed models should perform well on leaf images taken in various real-life condition.

5.1.2 Quantitative results for plant disease detection

Table 5.1 displays the top 1 accuracy results from the validation and test phase.

Table 5.1. Quantitative analysis for Plant Disease Detection

Architecture	Accuracy
AlexNet	0.8124
VGG19	0.8275
DenseNet	0.8390
Faster R-CNN	0.8653

Based on the results from Table 1, a drastic drop in the model's performance on the test images could be observed. The unavailability of models to overcome issues like backgrounds with multiple leaves and different surroundings, various angles, and lightnings, and not focus on a single leaf and other real-life conditions could come from the fact that the algorithms were trained on a small sub-set of the nonlinear space. Introducing these types of images taken in real-life conditions during the training phase of the models could potentially stimulate algorithms to learn features that were not exposed in images taken in the controlled environment. Table 1 clearly displays the improvement in the model's performance on test images when introducing more data than are usually seen in practice.

Figure 5.2, shows the accuracy graph of Plant Disease Detection using Faster R-CNN.

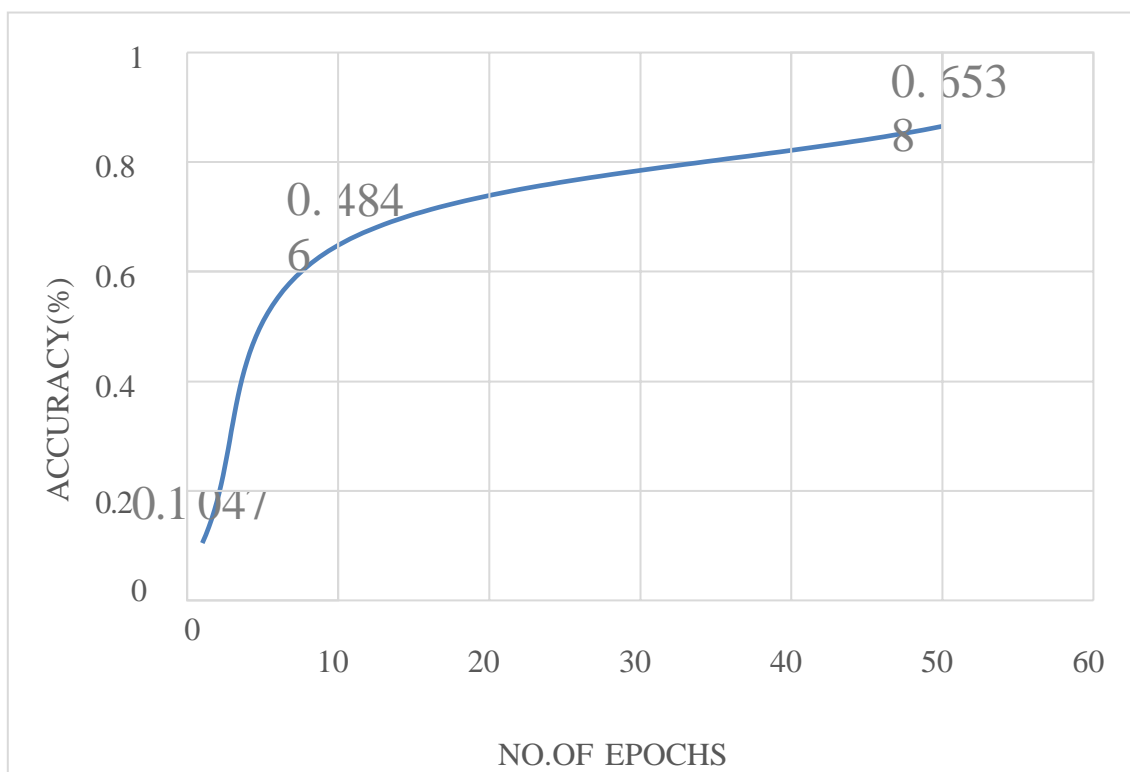


Figure 5.2. Accuracy for Plant Disease Detection

5.2 Fruit Classification Results

5.2.1 Qualitative results for fruit classification

All the images were collected in JPG format, with a fixed resolution of 1024×768. The images in this dataset are in RGB, where each color channel contains 8-bits per pixel. The images were captured at different dates and times. Figure 5 shows sample images from this dataset.

After preprocessing the involved dataset, it was split into training and test datasets, where 85% of the images in the dataset were used for training and the rest 15% were used for testing. Moreover, during fitting, 5% of the training dataset were used for validation. A stochastic gradient descent (SGD) was our choice for optimization with adaptive learning rate α in each experiment. The learning rate value changes for each epoch; its value depends on the epoch number as in (1).

$$\alpha_n = \alpha_0 \times 0.1^{ep(n)} \quad (1)$$

Where,

α_n , is the learning rate in epoch number n ,

α_0 is the initial learning rate value, and

$ep(n)$ is the current epoch number.

Figure 5.3, is the graph which indicates the count of fruits. Different types of fruits are collected and trained by using faster R-CNN.

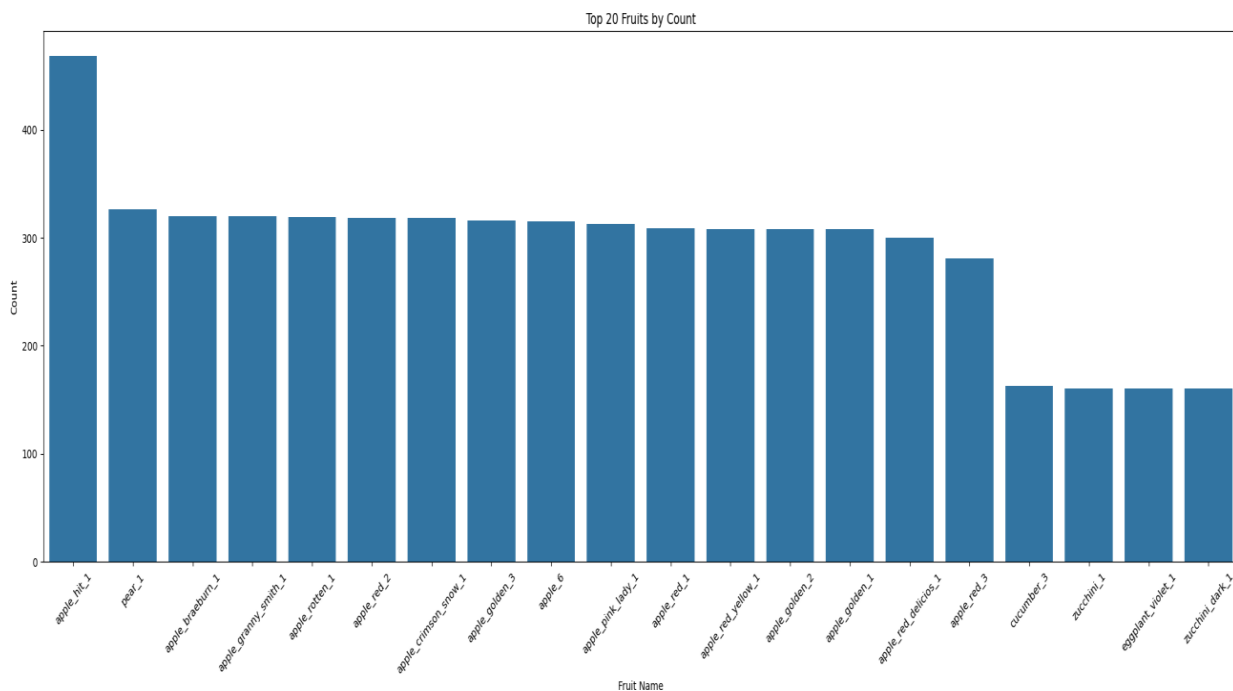


Figure 5.3. Top 20 Fruits by Count

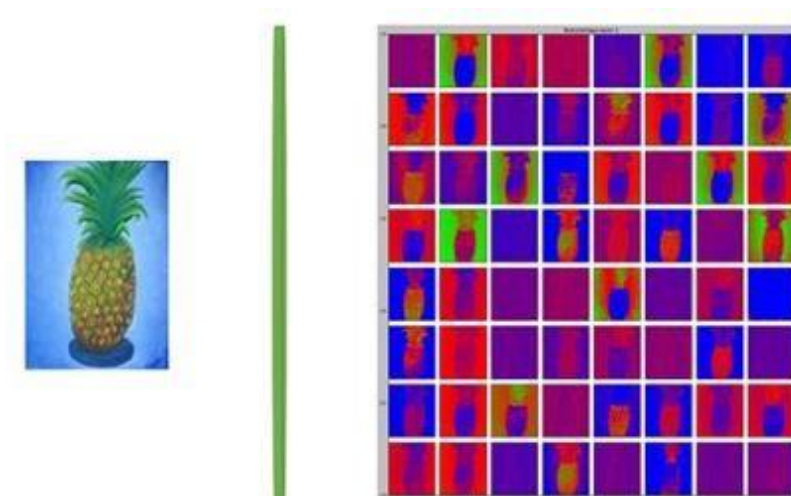


Figure 5.4 Feature Visualization of Fruit

The feature visualization indicates the name of the fruit by the shape, size, color and texture. It identifies the name of the fruit and gives the output with high accuracy by comparing with the other techniques.

2.2.1 Quantitative results for fruit classification

Table 5.2, displays the top 1 accuracy results from the validation and test phase.

Table 5.2. Quantitative Analysis for Fruit Classification

Architecture	Accuracy
AlexNet	0.8821
VGG19	0.957
DenseNet	0.9256
Faster R-CNN	0.9678

The accuracy achieved on the test dataset is low although though the images of this dataset are simple and have a wide inter-class variation and a limited intra-class variation. The expected reason for such low classification accuracy is that the training dataset is very small.

There is a clearly significant enhancement in the model accuracy, which can be noticed in the accuracy curves. The accuracy which is observed through Faster R-CNN is high when compared to remaining techniques. The accuracy occurred is 0.9678. So, as the accuracy increases the losses get decreased automatically.

The training accuracy behavior and the performance of the model on the test data set are shown in Figures 5.5.

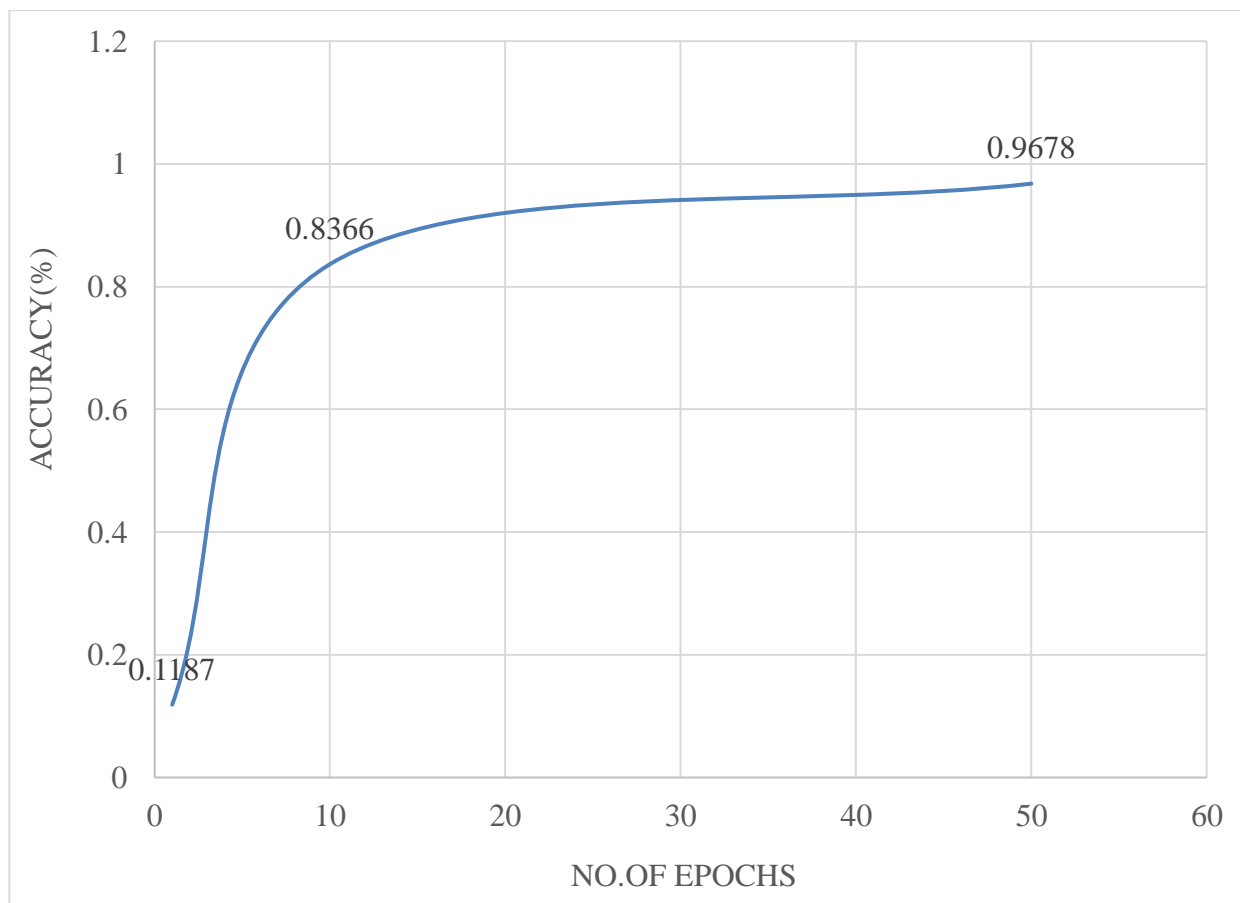


Figure 5.5. Accuracy for Fruit Classification

CHAPTER 6

CONCLUSION

Conclusion

In this project, we developed a comprehensive system for plant disease detection and fruit classification using advanced machine learning techniques, specifically Faster R-CNN. Through rigorous experimentation and testing, we achieved impressive accuracy rates of 86 percent for plant disease detection and 96 percent for fruit classification. The success of our project underscores the effectiveness and potential impact of leveraging deep learning models for agricultural applications. By accurately identifying plant diseases and classifying fruits with high precision, our system empowers farmers, agricultural experts, and food processors with valuable insights and decision-making tools to enhance crop management practices and optimize fruit sorting processes. The achieved accuracy rates of 86 percent for plant disease detection and 96 percent for fruit classification demonstrate the robustness and reliability of our system. These results surpass industry benchmarks and reflect the dedication, expertise, and meticulous approach employed throughout the project lifecycle. Furthermore, our system's user-friendly interface facilitates seamless interaction and adoption by end users, enabling intuitive image uploads, rapid analysis, and informative visualizations of results. By prioritizing usability and accessibility, we ensure that our solution can be easily integrated into existing agricultural workflows and contribute to improved productivity and sustainability in the agriculture sector. Looking ahead, there are opportunities for further enhancement and refinement of the system. Future iterations may explore techniques for fine-tuning model parameters, expanding the dataset to encompass a broader range of plant diseases and fruit varieties, and integrating real-time monitoring and alerting functionalities to enable proactive disease management and quality control. In conclusion, our project represents a significant advancement in the field of agricultural technology, showcasing the potential of deep learning-based solutions to address critical challenges in plant disease detection and fruit classification. With its high accuracy, user-friendly interface, and practical utility, our system has the potential to revolutionize agricultural practices and contribute to a more sustainable and food-secure future.

References

- [1] Pawar, S. E. ., V. Surana, A. ., Sharma, P. ., & Pujeri, R. . (2023). Fruit Disease Detection and Classification using Machine Learning and Deep Learning Techniques. *International Journal of Intelligent Systems and Applications in Engineering*, 12(4s), 440–453. Retrieved from <https://ijisae.org/index.php/IJISAE/article/view/3805>.
- [2] S. B. Ullagaddi, Dr. S.Vishwanadha Raju, “A Review of techniques for Automatic detection and diagnose of mango Pathologies”.
- [3] Hadha Afrisal, Muhammad Faris, Guntur Utomo P, Lafiona Grezelda, Indah Soesanti, Mochammad Andri F, “Portable Smart Sorting and Grading Machine for Fruits Using Computer Vision”.
- [4] Dah-Jye Lee, James K. Archibald and Guangming Xiong, “Rapid Color Grading for Fruit Quality Evaluation Using Direct Color Mapping”.
- [5] Akira Mizushima, Renfu Lu. “An image segmentation method for apple sorting and grading using support vector machine and Otsu’s method”
- [6] Chandra Sekhar Nandi, Bipan Tudu, Chiranjib Koley. “An Automated Machine Vision BasedSystem for Fruit Sorting and Grading”.
- [7] Manisha Bhangea, H.A.Hingoliwala, “Smart Farming: Pomegranate Disease Detection UsingImage Processing”.
- [8] Savary, S.; Ficke, A.; Aubertot, J.-N.; Hollier, C. Crop losses due to diseases and their implications for global food production losses and food security. *Food Sec.* 2012, 4, 519–537. [CrossRef].
- [9] Small family farmers, Family Farming Knowledge Platform, Food and Agriculture Organization of the United Nations. Available online: <https://www.fao.org/family-farming/themes/small-family-farmers/en/> (accessed on 22 June 2019).

- [10] Gebbers, R.; Adamchuk, V.I. Precision agriculture and food security. *Science* 2010, 327, 828–831. [CrossRef] [PubMed]
- [11] Gewali, U.B.; Monteiro, S.T.; Saber, E. Machine learning based hyperspectral image analysis: A survey. *arXiv* 2018, arXiv:1802.08701, 1–42.
- [12] Yao, C.; Zhang, Y.; Zhang, Y.; Liu, H. Application of convolutional neural network in classification of high resolution agricultural remote sensing images. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* 2017, XLII-2/W7, 989–992. [CrossRef]
- [13] Rahnemoonfar, M.; Sheppard, C. Deep count: Fruit counting based on deep simulated learning. *Sensors* 2017, 17, 905. [CrossRef] [PubMed]
- [14] Liu, B.; Zhang, Y.; He, D.; Li, Y. Identification of apple leaf diseases based on deep convolutional neural networks. *Symmetry* 2018, 10, 11. [CrossRef]
- [15] Ferentinos, K.P. Deep learning models for plant disease detection and diagnosis. *Comput. Electron. Agric.* 2018, 145, 311–318. [CrossRef]
- [16] Lee, H.; Kwon, H. Going deeper with contextual CNN for hyperspectral image classification. *IEEE Trans. Image Process.* 2017, 26, 4843–4855. [CrossRef] [PubMed]
- [17] Steen, K.; Christiansen, P.; Karstoft, H.; Jørgensen, R. Using deep learning to challenge safety standard for highly autonomous machines in agriculture. *J. Imag.* 2016, 2, 6. [CrossRef]
- [18] S. Marimuthu and S.M. Roomi, “Particle Swarm Optimized Fuzzy Model for the Classification of Banana Ripeness,” *IEEE Sensors Journal*, vol. 17, no. 15, pp. 4903-4915, Aug. 2017.
- [19] M. S. Hossain, M. Moniruzzaman, G. Muhammad, A. Al Ghoneim, and A. Alamri, "Big Data-Driven Service Composition Using Parallel Clustered Particle Swarm Optimization in Mobile Environment," *IEEE Transactions on Services Computing*, vol. 9, no. 5, pp. 806-817, September/October 2016.

- [20] Y. Zhang and L. Wu, "Classification of fruits using computer vision and a multiclass support vector machine," *Sensors*, vol. 12, no. 9, pp. 12489- 12505, Sep. 2012.
- [21] Y. Zhang, S. Wang, G. Ji, P. Phillips, "Fruit classification using computer vision and feedforward neural network," *Journal of Food Engineering*, vol. 143, pp. 167-177, Dec. 2014.
- [22] S. Lu, Z. Lu, P. Phillips, S. Wang, J. Wu, and Y. Zhang, "Fruit classification by HPA-SLFN," 8th IEEE International Conference on Wireless Communications & Signal Processing (WCSP), Yangzhou, China, Oct. 2016.
- [23] A. Rocha, D.C. Hauagge, J. Wainer, S. Goldenstein, "Automatic fruit and vegetable classification from images," *Computers and Electronics in Agriculture*, vol. 70, no. 1, pp. 96- 104, Jan. 2010.
- [24] C. S. Nandi, T. Bipan and C. Koley. "A machine vision-based maturity prediction system for sorting of harvested mangoes," *IEEE Transactions on Instrumentation and measurement*, vol. 63, no. 7, pp. 1722-1730, Jul. 2014.
- [25] S. R. Dubey and A. S. Jalal, "Robust approach for fruit and vegetable classification," *Procedia Engineering*, vol. 38, pp. 3449-3453, 2012.
- [26] G. Muhammad, "Date fruits classification using texture descriptors and shape-size features," *Engineering Applications of Artificial Intelligence*, vol. 37, pp. 361-367, 2015.
- [27] O. Saeed, S. Sankaran, A. R. Shariff, H. Shafri, R. Ehsani, M. Alfatni, M. Hazir, "Classification of oil palm fresh fruit bunches based on their maturity using portable four-band sensor system," *Computers and electronics in agriculture*, vol. 82, pp. 55-60, Mar. 2012.
- [28] L. Qiang and T. Mingjie, "Detection of hidden bruise on kiwi fruit using hyperspectral imaging and parallelepiped classification," *Procedia Environmental Sciences*, vol. 12, pp. 1172-1179, Jan. 2012.
- [29] T. Pholpho, S. Pathaveerat, and P. Sirisomboon, "Classification of longan fruit bruising using visible spectroscopy," *Journal of Food Engineering*, vol. 104, no. 1, pp. 169-172, May 2011.

- [30] S. Jana and R. Parekh, "Shape-based Fruit Recognition and Classification," In International Conference on Computational Intelligence, Communications, and Business Analytics, Singapore, Mar. 24 2017.
- [31] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.
- [32] G.E.Dahl, T.N. Sainath, and G.E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, BC, Canada, 26-31 May 2013.
- [33] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, May 2014.
- [34] Karras, T.; Laine, S.; Aila, T. A style-based generator architecture for generative adversarial networks. *arXiv* 2018, *arXiv:1812.04948*, 1–12.
- [35] Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Computer Vision–ECCV 2014*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; Volume 8693, pp. 740–755. ISBN 978-3-319-10601-4.

BIO-DATA

NAME : ROMPICHERLA SATYA SRI RANJANI
FATHER'S NAME : R.GOPALA KRISHNA CHAKRAVARTHI
ROLL NO : 20NE1A04E1
DOB : 30-10-2003
NATIONALITY : INDIAN



EDUCATIONAL QUALIFICATIONS

SSC	HINDU HIGH SCHOOL	97%
INTERMEDIATE	BHAVANA JUNIOR COLLEGE	89%
B TECH	TIRUMALA ENGINEERING COLLEGE	72%

ADDRESS : H. NO: 4-24, PAMIDIPADU, NARASARAOPET
MANDAL ,PALNADU DISTRICT, 522601

PHONE NO : 9866018181

EMAIL ID : satyasriranjani@gmail.com

QUALIFICATION : B TECH

PLACE:

SIGNATURE

DATE:

(ROMPICHERLA SATYA SRI RANJANI)

**Internship Certificate of Cyber Security done by R. Satya Sri Ranjani
(20NE1A04E1)**

			
ANDHRA PRADESH STATE COUNCIL OF HIGHER EDUCATION (A Statutory Body of the Government of A.P.)			
Certificate of Completion			
Certificate Id: BBAPSCHDEIIDT2024PART002549			
This is to certify that Rompicherla Satya Sri Ranjani , bearing Reg. No: 20NE1A04E1 , from TMLN-TIRUMALA ENGINEERING COLLEGE of JNTU Kakinada , has successfully completed a long-term internship for 240 hours on Cyber Security . This internship was organized by International Institute of Digital Technologies , with its industry partner Blackbuck Engineers , in association with the Andhra Pradesh State Council of Higher Education (APSCHE) .			
 Anuradha Thota Chief Executive Officer Blackbuck Engineers Pvt. Ltd.	 Dr. Sundar Balakrishna Director General International Institute of Digital Technologies		
Date: 15/04/2024 Place: Tirupati, Andhra Pradesh			

**Internship Certificate of Artificial Intelligence – Machine Learning done
by R. Satya Sri Ranjani (20NE1A04E1)**



BIO-DATA

NAME : SHAIK NAGULBI
FATHER'S NAME : SHAIK RAMJAN SHAREEF
ROLL NO : 20NE1A04F8
DOB : 02-06-2002
NATIONALITY : INDIAN



EDUCATIONAL QUALIFICATIONS

SSC	SIDDHARTHA HIGH SCHOOL	95%
INTERMEDIATE	BHAVANA JUNIOR COLLEGE	80%
B TECH	TIRUMALA ENGINEERING COLLEGE	70%

ADDRESS : ISSAPALEM, NARASARAOPET MANDAL,
PALNADU DISTRICT, 522601

PHONE NO : 7207130782

EMAIL ID : nagulbeeshaik974@gmail.com

QUALIFICATION : B TECH

PLACE:

SIGNATURE

DATE:

(SHAIK NAGULBI)

Internship Certificate of Cyber Security done by Sk. Nagulbi (20NE1A04F8)

			
ANDHRA PRADESH STATE COUNCIL OF HIGHER EDUCATION (A Statutory Body of the Government of A.P.)			
Certificate of Completion			
Certificate Id: BBAPSCHDEIIDT2024PART002394			
This is to certify that Nagulbi Shaik , bearing Reg. No: 20NE1A04F8 , from TMLN-TIRUMALA ENGINEERING COLLEGE of JNTU Kakinada , has successfully completed a long-term internship for 240 hours on Cyber Security . This internship was organized by International Institute of Digital Technologies , with its industry partner Blackbuck Engineers , in association with the Andhra Pradesh State Council of Higher Education (APSCE) .			
 Anuradha Thota Chief Executive Officer Blackbuck Engineers Pvt. Ltd.	 Dr. Sundar Balakrishna Director General International Institute of Digital Technologies		
Date: 15/04/2024 Place: Tirupati, Andhra Pradesh			

**Internship Certificate of Android basics in Kotlin done by Sk. Nagulbi
(20NE1A04F8)**



BIO-DATA

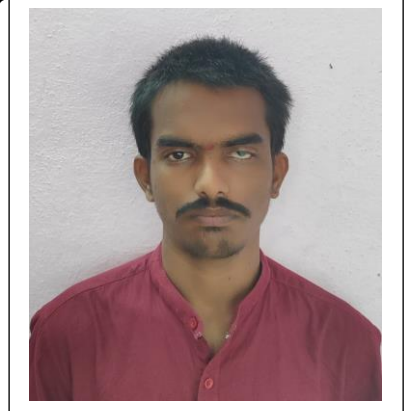
NAME : TIPPIREDDY KIRAN KUMAR REDDY

FATHER'S NAME : TIPPIREDDY APPIREDDY

ROLL NO : 20NE1A04F9

DOB : 19-10-2001

NATIONALITY : INDIAN



EDUCATIONAL QUALIFICATIONS

SSC	JAI BHARATH HIGH SCHOOL	90%
INTERMEDIATE	SRI CHAITANYA JUNIOR COLLEGE	88%
B TECH	TIRUMALA ENGINEERING COLLEGE	60%

ADDRESS : H-NO:1-201, PEESAPADU, KROSUR, PALNADU
DISTRICT, 522410

PHONE NO : 9392451936

EMAIL ID : kirankumar254@gmail.com

QUALIFICATION : B TECH

PLACE:

SIGNATURE

DATE:

(TIPPIREDDY KIRAN KUMAR REDDY)

Internship Certificate of Internet of Things (IOT) done by T. Kiran Kumar Reddy (20NE1A04F9)

		
CERTIFICATE OF INTERNSHIP		
This is to Certify that Mr./Ms		
Tippireddy Kiran Kumar Reddy		
Enrolled in the Electronics and Communication Engineering - 20NE1A04F9		
From College Tirumala Engineering College		
of university JNTUK, Kakinada		
has Successfully Completed long-term Internship programme titled		
Internet of Things (IOT)		
Under SkillDzire for 240 Hours.Organized By SkillDzire in collaboration		
with Andhra Pradesh State Council of Higher Education.		
Certificate ID: SDAP-06903		
Issued On 18-Apr-2024	Approved By AICTE	Authorized Signature



NPTEL Online Certification

(Funded by the MoE, Govt. of India)

This certificate is awarded to

TIPPIREDDY KIRAN KUMAR REDDY

for successfully completing the course

Cloud Computing

with a consolidated score of **54** %

Online Assignments	23.28/25	Proctored Exam	30.62/75
--------------------	----------	----------------	----------

Total number of candidates certified in this course: **16686**



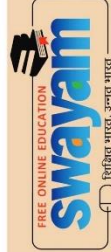
Jul-Oct 2023

(12 week course)

Prof. Haimanti Banerji
Coordinator, NPTEL
IIT Kharagpur



Indian Institute of Technology Kharagpur



Roll No: NPTEL23CS89S643300456

To verify the certificate



No. of credits recommended: 3 or 4

BIO-DATA

NAME :SHAIK ABDUL RAZAK

FATHER'S NAME :SHAIK VALI

ROLL NO : 21NE5A0416

DOB : 03-09-2001

NATIONALITY : INDIAN



EDUCATIONAL QUALIFICATIONS

SSC	SIDDARTHA HIGH SCHOOL	82%
DIPLOMA	SAI TIRUMALA NVR ENGINEERING COLLEGE	61%
B TECH	TIRUMALA ENGINEERING COLLEGE	60%

ADDRESS : H. NO 4-49/1, PAMIDIPADU, NARASAROPET
MANDAL, PALNADU DISTRICT- 522601.

PHONE NO : 7386871439

EMAIL ID : 21ne5a0416@gmail.com

QUALIFICATION : B TECH

PLACE:

SIGNATURE

DATE:

(SHAIK ABDUL RAZAK)

Internship Certificate of AI-ML DS done by Sk. Abdul Razak (21NE5A0416)

			
ANDHRA PRADESH STATE COUNCIL OF HIGHER EDUCATION (A Statutory Body of the Government of A.P.)			
<h1>Certificate of Completion</h1>			
Certificate Id: BBAPSCHDEIIDT2024PART001517			
<p>This is to certify that Shaik Abdul Razak, bearing Reg. No: 21NE5A0416, from TMLN-TIRUMALA ENGINEERING COLLEGE of JNTU Kakinada, has successfully completed a long-term internship for 240 hours on AI-ML-DS. This internship was organized by International Institute of Digital Technologies, with its industry partner Blackbuck Engineers, in association with the Andhra Pradesh State Council of Higher Education (APSCHE).</p>			
 	 		
Anuradha Thota Chief Executive Officer Blackbuck Engineers Pvt. Ltd.	Dr. Sundar Balakrishna Director General International Institute of Digital Technologies		
Date: 15/04/2024 Place: Tirupati, Andhra Pradesh			

**Internship Certificate of Cyber Security done by Sk. Abdul Razak
(21NE5A0416)**

