Rose color detection for blind people's using deep learning.

By
Name- Sajeeb Mandal
ID- 211-15-4045

FINAL YEAR DESIGN PROJECT REPORT

This Report Presented in Partial Fulfillment of the Requirements for the **Degree of Bachelor of Science in Computer Science and Engineering**

Supervised by

Ms. Nazmun Nessa Moon
Associate professor
Department of Computer Science and Engineering, Daffodil International University

Co-Supervised by

Ms. Faiza Feroz
Lecturer
Department of Computer Science and Engineering, Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

December 15, 2024

APPROVAL

This Project titled **"Rose color detection for blind people's using deep learning"**, submitted by Name, ID No: **211-15-4045** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 12 January, 2025.

BOARD OF EXAMINERS

Dr. Md. Zahid Hasan (ZH) — **Chairman**
Associate Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr. Md Alamgir Kabir (DMAK) — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Mr. Tanvirul Islam (TI) — **Internal Examiner**
Lecturer
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr. Md. Manowarul Islam — **External Examiner**
Associate Professor
Department of Computer Science and Engineering
Jagannath University

©Daffodil International University

DECLARATION

DECLARATION

We hereby declare that this project has been done by us under the supervision of **Ms. Nazmun Nessa Moon, Associate professor**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this projectnor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:

Ms. Nazmun Nessa Moon
Associate professor
Department of Computer Science and Engineering, Daffodil International University

Co-Supervised by:

Ms. Faiza Feroz (FFZ)
Lecturer
Department of Computer Science and Engineering, Daffodil International University

Submitted by:

12.01.25

Sajeeb Mandal
Student ID: 211-15-4045
Department of Computer Science and Engineering, Daffodil International University

©Daffodil International University

ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. We are deeply grateful to everyone who has assisted us in one way or another.

First, we express our heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for us to complete the **Final Year Design Project(FYDP)** successfully.

We are grateful and wish our profound indebtedness to **Ms. Nazmun Nessa Moon**, **Associate professor**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **Deep Learning** to carry out this project. His endless pa- tience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting themat all stages have made it possible to complete this project.

We would like to express our heartfelt gratitude to the Head of the Department of Com- puter Science and Engineering, for his kind help in finishing our project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

We would like to thank our entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

This thesis presents the development of a mobile app-based solution for rose color detection designed specifically for visually impaired individuals. The solution leverages advanced Vision Transformer (ViT) architectures, particularly ViT-B16 and ViT-B32, to enable real-time, accurate, and accessible color recognition. Addressing the limitations of traditional color identification methods, the proposed solution empowers users to independently experience and identify rose colors, fostering inclusivity and autonomy. The study incorporates synthetic data generation techniques to overcome the challenges of limited labeled datasets, enhancing model generalization across diverse environmental conditions. The lightweight nature of ViT-B16 and ViT-B32 ensures compatibility with standard mobile devices, optimizing computational efficiency while maintaining high accuracy. Intuitive feedback tailored to the needs of visually impaired users provides actionable and descriptive insights into detected colors. The research methodology involves data collection, model training using ViT-B16 and ViT-B32 architectures, and iterative app design, followed by rigorous testing under varied real-world conditions to evaluate performance. The results demonstrate the app's effectiveness, achieving 99.81% accuracy, and potential as a practical assistive tool. This study contributes to the growing field of AI-driven assistive technologies by addressing critical gaps in accessibility, dataset diversity, and real-world adaptability. Beyond its immediate application in rose color detection, the findings have broader implications for developing inclusive technologies that enhance the quality of life for individuals with visual impairments. The thesis concludes with recommendations for future improvements and scalability of the proposed solution.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

Agriculture, as a foundational pillar of human civilization, continues to play a vital role in the global economy by providing food, employment, and raw materials. Beyond consumable crops, ornamental plants like roses hold immense economic and cultural significance due to their aesthetic appeal and symbolic importance. Among the defining features of roses, colour stands out as a critical attribute, adding value and enabling differentiation among varieties. However, colour identification and appreciation can be challenging for individuals with visual impairments, limiting their ability to experience the rich symbolism and beauty of roses.

Technological advancements in machine learning (ML) and deep learning (DL) are paving the way for innovative solutions to address these challenges. Mobile applications equipped with intelligent algorithms now offer real-time colour detection capabilities, enabling visually impaired individuals to identify and interpret colours independently through smartphone cameras [1]. Modern deep learning architectures, such as Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs), have significantly enhanced the accuracy and scalability of image-based tasks, making them highly suitable for developing accessible solutions for rose colour detection [2].

While CNNs are adept at processing image data and extracting key features like colour and texture, ViTs leverage self-attention mechanisms to capture intricate details, allowing for precise identification of subtle colour variations [3]. Hybrid models combining CNNs and ViTs have emerged as a promising approach, balancing computational efficiency with high accuracy for mobile-based applications [3]. Additionally, lightweight models optimized for mobile and IoT devices are now

making real-time, app-based colour detection a practical reality, empowering visually impaired users with greater autonomy and interaction with their environment [9,10].

1.2 Motivation

For individuals with visual impairments, experiencing and identifying rose colours is a complex challenge due to the lack of accessible, efficient, and user-friendly tools. Traditional methods, such as relying on human assistance or specialized colour identification devices, are often impractical, expensive, and limited in scope. These barriers highlight the urgent need for innovative solutions that cater to the specific needs of this community, enabling them to interact more meaningfully with their surroundings.

Recent advancements in deep learning, including CNNs and ViTs, offer an unprecedented opportunity to bridge this gap. CNNs have proven effective in feature extraction for image-based tasks, as demonstrated in plant classification studies [4]. However, these models often require large, diverse datasets to achieve generalization and accuracy, which poses a significant limitation in the context of rose colour detection [5]. To address data scarcity, techniques like synthetic data generation using Generative Adversarial Networks (GANs) have been employed to augment training datasets and improve model robustness. For instance, the "LeafyGAN" model successfully enhanced classification tasks by creating varied synthetic samples, a strategy that could be adapted for rose colour detection [5].

ViTs, on the other hand, excel in capturing complex visual patterns and subtle distinctions, outperforming CNNs in tasks involving nuanced image features under diverse conditions [6,7]. This makes them particularly suitable for accurate rose colour identification. Recent research has also explored hybrid models that combine the strengths of CNNs for primary feature extraction with ViTs for refined classification, achieving high accuracy while maintaining compatibility with mobile platforms [3]. Such advancements highlight the potential for developing accessible and scalable solutions for real-world applications.

A key motivation for this research is to create a lightweight, mobile-compatible solution that addresses the unique needs of visually impaired users. By focusing on model interpretability and accessible feedback mechanisms, the aim is to deliver actionable insights that empower users to independently identify and experience rose colors. Additionally, the integration of efficient algorithms into mobile

applications ensures that such tools remain practical, cost-effective, and widely adoptable for this underserved demographic [9,13].

1.3 Objectives

This study aims to develop an accessible, efficient, and accurate mobile app for rose color detection, specifically tailored for visually impaired individuals. The app will leverage advanced machine learning models to provide real-time color recognition, allowing users to experience and identify rose colors independently. The specific objectives are as follows:

- To design a machine learning model capable of accurately detecting and identifying various rose colors under real-world conditions. The model will be optimized to handle diverse lighting, background, and environmental variations, ensuring reliability and precision.
- To incorporate augmentATION limited labelled datasets for rose colour detection. This approach aims to enhance model generalizability by creating diverse colour samples, addressing data scarcity and improving overall accuracy.
- To develop a computationally efficient, lightweight architecture suitable for real-time use on standard mobile devices. The model will balance high accuracy with minimal resource requirements, ensuring it is accessible to users with basic smartphones or IoT-enabled devices.
- To improve model interpretability and create a user-friendly interface that provides intuitive, descriptive feedback about detected colours. This objective will ensure those visually impaired users receive meaningful, easy-to-understand colour descriptions, making the app an effective assistive tool.

By fulfilling these objectives, this study aims to create a practical, accessible, and inclusive solution for rose color detection, offering visually impaired individuals a new way to experience and interact with the beauty of roses through a user-friendly mobile application.

1.4 Methodology

The proposed methodology for rose color detection employs a systematic approach to ensure accurate, efficient, and accessible results. It begins with the collection of a diverse rose color dataset, featuring images captured under varying lighting conditions and backgrounds. Preprocessing is then applied to the dataset, including normalization to scale pixel values for uniformity, resizing images to fixed

dimensions for compatibility with the model's input requirements, and adjusting the DPI to balance detail retention and computational efficiency. Advanced deep learning architectures, specifically pre-trained Vision Transformers (ViTs), are utilized for rose color detection. Various configurations of ViTs, such as ViT b16 (image sizes 32 and 64) and ViT b32 (image sizes 32 and 64), are evaluated to identify the best-performing model in terms of accuracy and computational efficiency. The optimal ViT model is then optimized and converted into TensorFlow Lite format (best_model_float32.tflite) to ensure seamless compatibility with mobile devices. A user-friendly mobile application, developed using Flutter, integrates the optimized model to provide real-time rose color detection. The application enables visually impaired users to capture rose images through their smartphone camera, with the detected color information delivered audibly for accessibility. This methodology combines state-of-the-art deep learning techniques with intuitive mobile application design, offering an inclusive and practical solution for rose color detection tailored to the needs of visually impaired individuals.

1.5 Project Outcome

The project aims to deliver an accessible and efficient rose color detection system that leverages advanced deep learning techniques and a user-friendly mobile application. By utilizing a diverse rose color dataset and applying rigorous preprocessing steps, the system ensures high-quality input data for training and testing. The integration of pre-trained Vision Transformers (ViTs) allows for accurate and robust color detection, with comparisons among various configurations (e.g., ViT b16 and ViT b32 with different image sizes) ensuring optimal model selection. The best-performing model is optimized and converted into a lightweight TensorFlow Lite format, enabling real-time functionality on mobile devices. The developed mobile application, built using Flutter, offers a seamless interface for visually impaired users, allowing them to capture rose images and receive real-time audio feedback on the detected colors. This innovative solution empowers visually impaired individuals with greater independence and interaction with their environment by bridging the gap between deep learning advancements and accessible technology.

1.6 Organization of the Report

This report is systematically organized into six chapters, each addressing critical aspects of the research on rose color detection for visually impaired users using

Vision Transformers (ViT), lightweight TensorFlow Lite models, and mobile application integration. The structure and content of the report are as follows:

Chapter 1: Introduction

This chapter introduces the research topic, articulating the motivation, objectives, and methodology of the study. It highlights the significance of rose color detection for visually impaired users, sets the context for the study, and outlines the expected outcomes of the project.

Chapter 2: Background

This chapter reviews the existing literature, focusing on related applications and technologies in color detection and deep learning. It identifies gaps in prior research, such as the lack of accessibility-focused solutions, justifying the need for this study.

Chapter 3: Research Methodology

This chapter details the methodology adopted for the study, describing the steps involved, including dataset preprocessing (e.g., normalization, resizing, and DPI adjustments), model selection using pre-trained ViTs, and lightweight optimization. Diagrams, such as the workflow and system architecture, illustrate the process, and the project plan outlines the development phases and task allocation.

Chapter 4: Implementation and Results

This chapter delves into the technical implementation of the optimized TensorFlow Lite model and its integration with the Flutter-based mobile application. It includes a comprehensive evaluation of the models, presenting comparisons of ViT configurations and the experimental results of the best-performing model. A critical discussion on model performance and usability is also provided.

Chapter 5: Engineering Standards and Design Challenges

This chapter examines the compliance of the research with relevant software and mobile application standards, focusing on accessibility features and ethical considerations. It also discusses societal and environmental impacts, challenges in designing lightweight and efficient models, and strategies employed to overcome these obstacles.

Chapter 6: Conclusion

The final chapter summarizes the findings and contributions of the study, reflecting on its implications for visually impaired users and the broader field of assistive technology. It also outlines potential future enhancements to improve accuracy, efficiency, and accessibility.

This structured organization ensures a logical progression of ideas, from introducing the research problem to discussing its outcomes and implications, providing a comprehensive understanding of the study's contributions and significance.

Chapter 2

Background

2.1 Introduction

The literature review chapter examines the advancements in machine learning and deep learning methodologies for color detection and their applications in assistive technologies. It highlights the use of Convolutional Neural Networks (CNNs), Vision Transformers (ViTs), and hybrid models for achieving accurate and efficient classification tasks, particularly in real-world scenarios. Special attention is given to the role of lightweight architectures and synthetic data augmentation in addressing challenges such as dataset scarcity, computational efficiency, and model adaptability.

The chapter also explores relevant studies focusing on assistive applications, emphasizing the need for user-centric design and seamless integration with mobile devices. By analyzing the strengths, limitations, and contributions of existing works, the review establishes a theoretical foundation and identifies gaps that this thesis aims to address.

2.2 Literature Review

Roses, with their rich colour diversity and cultural symbolism, hold significant value in ornamental horticulture and agriculture. Colour plays a vital role in distinguishing rose varieties and assessing their quality, especially for individuals with visual impairments who rely on external aids to experience colour. Recognising this, researchers have begun to explore machine learning (ML) and deep learning (DL) models for mobile-based solutions in colour detection, which are both efficient and adaptable for real-time use. Current literature reflects the growing interest in advanced architectures, such as Vision Transformers (ViTs) and Convolutional Neural Networks (CNNs), for colour

classification tasks across various settings, highlighting their potential in mobile-based rose colour detection.

Saini [1] demonstrated the efficacy of ViTs for fine-grained feature extraction, achieving high accuracy rates in classification tasks through optimal training and stopping strategies. Their model, with an accuracy of 99.86%, showcased how ViTs capture subtle features, which is critical in colour detection applications. Their work underscores ViTs' potential in recognising the detailed colour gradations necessary for accurate mobile-based rose colour detection.

Khaleel [4] applied CNNs to classify leaf diseases in roses, revealing CNNs' ability to manage image-based classification effectively. Although disease detection differs from colour detection, the study's robust pre-processing and real-field data capture methods such as resizing and augmentation, can inform similar workflows in color classification applications. Practical applications in mobile colour detection could follow their approach, capturing consistent images under varying environmental conditions to ensure reliable colour recognition.

Comparative studies on non-plant datasets also offer insights into rose colour detection. For instance, You [14] proposed a lightweight skin segmentation model that used advanced data augmentation techniques to improve model generalizability without depending heavily on colour information. Such techniques could be instrumental in designing mobile apps that perform reliably under diverse lighting conditions, a critical factor for colour-based rose classification.

Sawarkar and Kawathekar [15] reviewed digital image processing methods for rose disease detection, detailing steps such as image segmentation and feature extraction. Although focused on disease detection, these foundational image processing methods could enhance mobile-based colour detection workflows, facilitating real-time processing on low-power devices. This highlights the potential for hybrid approaches that integrate classical image processing with DL to optimise mobile application performance.

Kusuma and Jothi's [16] work on betel leaf classification further emphasises DL's applicability in plant analysis. Their study achieved over 98% accuracy using models like DenseNet201, underscoring the effectiveness of dense architectures in detailed classification tasks. The precision required in classifying betel leaf characteristics could parallel that of identifying subtle differences in rose colours, making DenseNet or similar architectures viable for mobile-based colour detection.

In the domain of plant pathology, Thai [17] explored a ViT-based model for cassava leaf disease detection, incorporating techniques such as Least Important Attention Pruning

(LeIAP) to streamline model performance. Optimising model complexity is essential for mobile applications, where resource efficiency is paramount. This research suggests that pruning and other model compression techniques could ensure high performance in a mobile colour detection app without compromising accuracy.

Ahmed [6] reviewed hybrid models that combine CNNs and ViTs, showing that the integration of these architectures improves classification accuracy across diverse datasets. In a mobile app for rose colour detection, a hybrid CNN-ViT model could potentially enhance colour recognition accuracy, combining CNN's feature extraction capabilities with ViT's ability to capture detailed colour distinctions.

Bhowmik [8] tailored a ViT model for Java Plum disease detection, achieving high accuracy through careful tuning of model hyperparameters. This study highlights the importance of hyperparameter optimization in maximizing model accuracy, an approach relevant to mobile applications where colour variations can be nuanced and require precise recognition.

Further reviews, such as those by Mustofa [5] have outlined deep learning models for plant disease detection, evaluating different architectures for scalability and efficiency. Models such as YOLO and other lightweight architectures are particularly relevant to mobile applications, where fast and accurate colour detection is essential for real-time use, especially when the device has limited processing power.

Thai [17] demonstrated ViTs' applicability in mobile-based applications through their work on cassava disease detection. By deploying ViTs on edge devices, they showcased the model's potential for real-time diagnostics in resource-constrained settings, suggesting that similar ViT applications could enable rose colour detection on mobile devices, offering individuals with visual impairments a user-friendly, accurate solution.

Singh [11] developed "LeafyGAN," a GAN-based model to augment plant disease datasets, effectively addressing the challenges of small, imbalanced labelled datasets. By generating synthetic data, LeafyGAN created enhanced training samples, enabling a MobileViT model to reach an accuracy of 99.92% on the PlantVillage dataset. This study underscores the potential of GAN-augmented data in improving model robustness for applications like rose colour detection, where diverse colour samples are essential for reliable mobile-based classification.

Barman [12] introduced "Vit-SmartAgri," a ViT-based smartphone application designed for tomato disease detection, highlighting the model's mobile compatibility and lightweight design. Achieving an accuracy of 90.99%, Vit-SmartAgri outperformed Inception V3, demonstrating how ViTs can enhance mobile applications with minimal

computational resources. Their approach suggests that similar lightweight ViT models could be adapted for mobile-based rose colour detection, ensuring accessible, on-the-go colour classification.

Keerthan Bhat [2] proposed "LeafViT," a ViT model that uses self-attention mechanisms to identify subtle features, achieving 97.15% accuracy on the Plant Pathology 2020 dataset. The model highlights ViT's capability to capture fine-grained visual details, which is essential in colour detection tasks. Data augmentation and balancing strategies used in LeafViT further underscore the potential of ViTs to improve robustness in mobile applications where colour recognition precision is vital.

Thakur [3] introduced a hybrid CNN-ViT model for plant disease detection, which achieved high accuracy across large and small datasets with a low parameter count. Their work demonstrates the effectiveness of combining CNN's feature extraction strengths with ViT's classification capabilities, offering a resource-efficient model suitable for IoT and mobile applications.

Bhuyan and Singh [7] conducted a comparative study of CNNs and ViTs on various leaf disease datasets, demonstrating that ViTs consistently outperformed CNNs on challenging datasets. With a ViT-30 model achieving over 98% accuracy on the Rice Leaf dataset, the authors provide insights into optimal ViT configurations for visual tasks, such as colour detection, which require precise feature sensitivity.

Hossain et al. [10] optimised a ViT model for mango leaf disease detection using a pre-trained Data-efficient Image Transformer (DeiT), achieving 99.75% accuracy with reduced training requirements. Their integration of a mobile backend further illustrates ViT's adaptability to real-time applications, suggesting its viability for mobile-based rose colour detection, particularly for colour-based classification tasks that demand high accuracy.

Gangwar [9] developed a multi-model Convolution Vision Transformer (CVT) for tomato disease detection, designed specifically for varied backgrounds and minimal storage requirements. Achieving high accuracy with reduced training time, Gangwar's approach supports the idea that compact, resource-efficient models are ideal for mobile and IoT platforms, emphasizing their application in color-based rose classification.

Sundaraj [13] applied a ViT model in precision agriculture, achieving 94% precision in plant disease classification. Their work, which focuses on diverse datasets of diseased and healthy plants, underscores ViTs' potential to support mobile-based solutions that enable quick, accurate disease or color recognition in agriculture. The findings advocate for ViT's role in enhancing digital agricultural diagnostics through mobile apps.

Chen [19] proposed a hybrid CNN-Transformer model for tomato leaf disease detection, utilizing a CycleGAN-based data augmentation method to handle class imbalance. Their model achieved high accuracy across multiple datasets, demonstrating the versatility of hybrid models for applications requiring generalizability and robustness, such as rose color detection for users in varied lighting conditions.

Ahmed [20] implemented a ViT model for tea leaf disease detection, achieving 99.53% accuracy. This model, which was trained on diverse datasets, exemplifies how ViTs enhance agricultural diagnostic accuracy. In the context of rose color detection, a similar ViT-based model could ensure that subtle color distinctions are captured effectively, making it suitable for real-time mobile deployment.

Lastly, Salamai [21] introduced a lesion-aware ViT network for paddy disease detection, achieving 98.74% accuracy by capturing both local and global contextual features. Their work demonstrates how ViT-based systems can enhance precision in agricultural diagnostics, supporting applications in colour detection where nuanced colour variations must be accurately recognized.

Table 2.1: Summary of Literature Reviewed.

Author (Year)	Models Used	Accuracy	Dataset Information
Saini, Guleria, & Sharma (2024)	ViT	99.86%	Rose Leaf Dataset
Khaleel et al. (2022)	CNN	N/A	Real Field and Existing Dataset
You et al. (2023)	Lightweight Skin Segmentation Model	94.92% F-score	Skin Dataset
Sawarkar & Kawathekar (2018)	Image Processing Techniques	N/A	Rose Plant Dataset
Kusuma & Jothi (2024)	VGG19, DenseNet201, ResNet152V2, ViT	98.77%	Betel Leaf Dataset

Thai, Le, & Nguyen (2023)	ViT with LeIAP, SPMM	N/A	Cassava Leaf Dataset
Ahmed et al. (2023)	ICVT, GreenViT, DenseNet, ResNet-50V2, YOLO, CNN	Varied	Tea Leaf Datasets
Bhowmik et al. (2024)	Optimized ViT	97.51%	Java Plum Dataset
Mustofa et al. (2023)	ViT, DCNN, CNN, YOLO	Varied	Public Leaf Disease Datasets
Thai et al. (2021)	ViT	N/A	Cassava Leaf Dataset
Singh et al. (2024)	LeafyGAN, MobileViT	99.92%	PlantVillage and PlantDoc
Barman et al. (2024)	ViT, Inception V3	90.99%	Tomato Leaf Dataset
Keerthan Bhat et al. (2022)	ViT	97.15%	Plant Pathology 2020 Dataset
Thakur et al. (2023)	ViT-CNN Hybrid	98.86%	PlantVillage, Embrapa
Bhuyan & Singh (2024)	ViT, CNN	98.41%	Rice, Tea, and Maize Leaf Datasets
Hossain et al. (2024)	ViT (DeiT)	99.75%	Mango Leaf Dataset
Gangwar et al. (2024)	CVT	93.51%	Tomato Leaf Dataset (Varied Backgrounds)
Sundaraj et al. (2024)	ViT	94%	Diverse Plant Leaf Dataset
Chen et al. (2024)	CycleGAN, Transformer, CNN	99.45%	PlantVillage, Private Dataset

Ahmed et al. (2023)	Fuzzy-Based ViT	99.53%	Tea Leaf Dataset

2.2.1 Similar Applications

Several research studies and applications have explored methodologies relevant to rose color detection and similar tasks in plant and agricultural diagnostics. For instance, Saini et al. (2024) demonstrated the effectiveness of Vision Transformers (ViTs) in fine-grained feature extraction for classification tasks, achieving an accuracy of 99.86% on the Rose Leaf Dataset. Their work highlights ViTs' potential for applications requiring detailed recognition, such as rose color detection.

Khaleel et al. (2022) used Convolutional Neural Networks (CNNs) to classify diseases in roses, showcasing CNNs' capacity for image-based classification. While their work focused on disease detection, their robust preprocessing techniques and real-field data collection methods can inform workflows for mobile-based color detection.

Applications outside the domain of plants also provide valuable insights. You et al. (2023) introduced a lightweight skin segmentation model with advanced data augmentation to improve generalizability. This approach can be adapted to rose color detection apps, ensuring accuracy under varied lighting conditions. Similarly, Barman et al. (2024) developed a ViT-based smartphone application for tomato disease detection, achieving 90.99% accuracy while maintaining mobile compatibility, offering a framework for real-time mobile applications in agriculture.

Lastly, Singh et al. (2024) employed a GAN-augmented MobileViT model to address imbalanced datasets, achieving 99.92% accuracy. Their approach to generating diverse and high-quality data is particularly relevant for rose color detection, where variations in hue and lighting need careful handling.

2.2.2 Related Research

The research literature on machine learning and deep learning applications for plant diagnostics demonstrates significant advancements in accuracy and efficiency. Sawarkar and Kawathekar (2018) highlighted the importance of image preprocessing techniques like segmentation and feature extraction, foundational steps for color detection workflows.

Kusuma and Jothi (2024) explored advanced deep learning architectures, achieving 98.77% accuracy in betel leaf classification. Their findings suggest that models like

DenseNet201 could effectively handle nuanced tasks such as detecting subtle color differences in roses.

Hybrid models combining CNNs and ViTs have also shown promise. Ahmed et al. (2023) reviewed hybrid approaches, demonstrating that integrating CNNs' feature extraction with ViTs' fine-grained classification improves performance across datasets. Similarly, Thakur et al. (2023) achieved 98.86% accuracy using a CNN-ViT model, illustrating the potential for efficient and precise classification in resource-constrained settings, such as mobile apps.

Techniques for optimizing ViTs for mobile applications are gaining traction. Thai et al. (2023) and Hossain et al. (2024) demonstrated ViTs' ability to balance high accuracy with lightweight deployment, making them suitable for real-time tasks like rose color detection. Thai et al. incorporated pruning techniques to streamline model performance, while Hossain et al. used Data-efficient Image Transformers (DeiT) for improved scalability and efficiency.

Overall, the literature emphasizes the effectiveness of advanced deep learning models and hybrid approaches for plant diagnostics, offering valuable insights for the development of mobile-based rose color detection applications.

2.3 Gap Analysis

The literature on detection using machine learning and deep learning showcases a wide range of methodologies and model architectures, including Vision Transformers (ViT), Convolutional Neural Networks (CNN), hybrid models, and data augmentation techniques using GANs. While these studies have achieved high accuracy across various datasets, several critical gaps remain that suggest areas for future research and improvement in practical agricultural applications.

- Many studies utilise well-structured datasets like PlantVillage, which contain images with clear backgrounds and minimal noise. Although this controlled setup allows models to achieve high accuracy, it limits the model's adaptability to real-world conditions, where leaf images may contain varied backgrounds, lighting variations, and noise. Only a few studies, such as Gangwar et al. (2024), focus on detecting diseases in images with complex backgrounds, highlighting a need for more research on dataset diversity to ensure model robustness in diverse agricultural settings.

- Most high-performing models require extensive labelled datasets, which are often unavailable for many plant species or disease types, especially in regions where agriculture is a primary livelihood but digital resources are limited. Studies such as Singh et al. (2024) attempted to address this with GAN-based synthetic data generation (e.g., LeafyGAN), but there is still a scarcity of approaches that systematically address data limitations across various plant diseases. Further research on synthetic data generation and transfer learning could improve model accuracy in low-data environments.
- Despite the growing use of lightweight models, such as MobileViT and hybrid CNN-ViT architectures, many models are still computationally intensive and unsuitable for deployment on low-end devices accessible to farmers. Thakur et al. (2023) and Gangwar et al. (2024) achieved lower model sizes, but more studies are needed to make high-accuracy models feasible on mobile and IoT-enabled devices. Developing efficient, low-parameter models for real-time, field-deployable applications remains a key area for future research.
- While some studies, such as Thakur et al. (2023) and Gangwar et al. (2024), explore the use of IoT and mobile integration, the majority do not address how these models can integrate into broader precision agriculture systems. Limited exploration into data fusion from IoT sensors, such as environmental conditions, could enhance model accuracy and provide more comprehensive disease diagnostics. Future research could examine how models for plant disease detection might incorporate additional data streams to deliver holistic insights into plant health.

While recent advancements in ViT and hybrid architectures have improved detection accuracy, addressing these gaps in dataset diversity, low-data adaptability, computational efficiency, multi-crop generalisation, IoT integration, and model interpretability would enhance the practical usability and scalability of these models. A holistic approach that incorporates real-world adaptability, efficient deployment, and improved explainability could significantly contribute to the effectiveness of plant disease management in agriculture.

2.4 Summary

This chapter provides an in-depth analysis of existing research on machine learning and deep learning techniques for color detection and assistive technologies. It underscores the strengths of CNNs in feature extraction and the superior performance of ViTs in capturing fine-grained visual details. The use of hybrid CNN-ViT architectures and

synthetic data augmentation techniques is presented as a promising approach to overcome data limitations and improve model robustness.

The chapter identifies gaps in existing studies, such as the lack of diverse datasets, real-world adaptability, and accessibility features tailored for visually impaired users. These insights justify the need for a mobile-based rose color detection solution that addresses these challenges while emphasizing inclusivity and sustainability. This review forms the basis for the methodologies and design considerations outlined in the subsequent chapters.

Chapter 3

Research Methodology

3.1 Methodology

3.1.1 Overview

The methodology of this study is centered on creating an accessible solution for rose color detection to assist visually impaired individuals. The process begins with the preparation of a comprehensive dataset, Rose_Data, which combines original rose images with additional samples collected directly from natural environments. This diverse collection of images ensures that the model can generalize effectively, accommodating various lighting conditions and backgrounds to provide accurate color detection results.

Following dataset preparation, a series of preprocessing steps are applied to standardize and optimize the images for model training. These steps include normalization of pixel values to a specified range for consistency, resizing images to uniform dimensions to meet model input requirements, and adjusting DPI (dots per inch) to retain quality while optimizing performance. These preprocessing techniques prepare the data for efficient processing by deep learning models, laying a strong foundation for effective color classification

The deep learning component of the methodology involves experimenting with multiple pretrained Vision Transformers (ViTs) to identify the most effective model for rose color detection. This stage includes comparing different ViT architectures with variations in image resolution and configurations, such as ViT b16 and ViT b32. By evaluating multiple model architectures, this approach ensures the selection of a model that combines accuracy with computational efficiency, making it suitable for deployment on mobile devices.

Once the optimal model is identified, it is converted into a TensorFlow Lite (.tflite) format for compatibility with mobile applications. The final user-facing application, developed using the Flutter framework, provides an intuitive interface with a "Detect" button to initiate color recognition. The detected color is displayed visually and conveyed audibly through integrated audio output, allowing visually impaired users to independently identify rose colors. This combination of machine learning, deep learning, and mobile accessibility enhances the system's usability and empowers end-users with greater independence.

3.1.2 Proposed Methodology/ System Design

The methodology depicted outlines a system for rose color detection using deep learning techniques. It begins with an original rose dataset, which undergoes preprocessing steps such as normalization, image resizing, and setting the DPI to standardize the input data. The preprocessed images are then fed into deep learning models, specifically pre-trained Vision Transformer (ViT) architectures. Various ViT models are compared based on configurations such as patch size (b16, b32) and image input dimensions (32, 64) to determine the most effective model. The best-performing model is converted into a TensorFlow Lite format (best_model_float32.tflite) for deployment. This lightweight model is integrated into a Flutter-based mobile application, which allows end users to upload rose images for detection. The app provides feedback through audio output and visual reports, making it accessible and user-friendly.

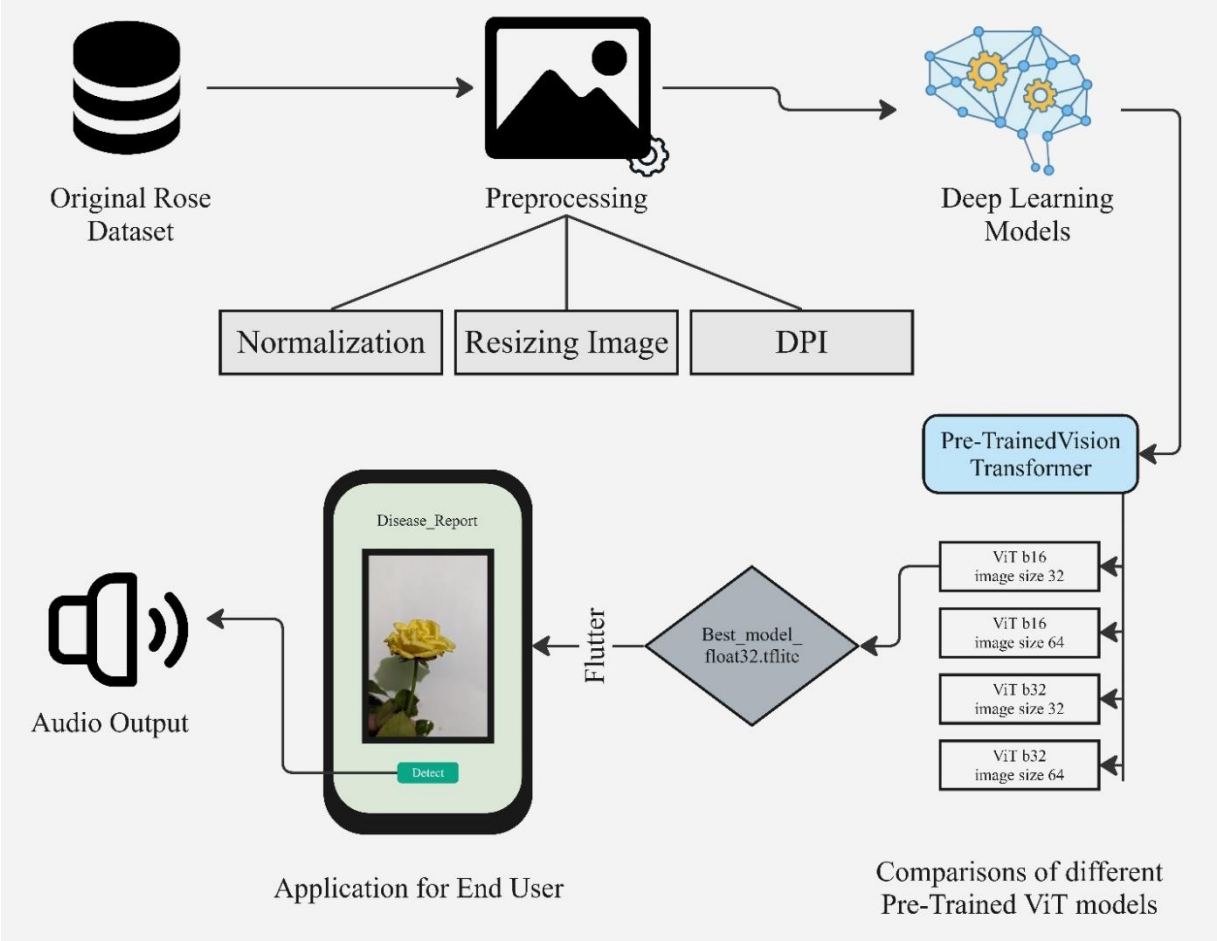


Figure 3.1: This is a sample methodology diagram

3.1.3 Functional and Nonfunctional Requirements

The system for rose color detection is designed to meet several functional and non-functional requirements. Functionally, it must accurately classify rose colors into predefined categories such as red, pink, yellow, and white, ensuring high precision in real-time processing. The application should preprocess input images by performing tasks like resizing, normalizing, and augmenting to handle environmental variations such as lighting and angles. A machine learning model, such as a Vision Transformer (ViT), CNN, or a hybrid CNN-ViT, will be integrated to ensure robust color detection. The mobile application will feature an intuitive user interface, allowing users to capture images easily and view the detected color. Additionally, the app will provide audio feedback for visually impaired users, enhancing accessibility.

On the non-functional side, the system must be resource-efficient, ensuring smooth performance on mobile devices with limited computational power. It should be scalable to support additional color categories or features in the future. The application must maintain high reliability, ensuring consistent color detection across diverse environmental conditions, and should operate with minimal latency for a seamless user experience. Moreover, it should adhere to modern design principles, ensuring usability, security, and compatibility with popular mobile platforms like Android and iOS.

3.1.4 Context Diagram

3.1.5 Data Flow Diagram Level 1

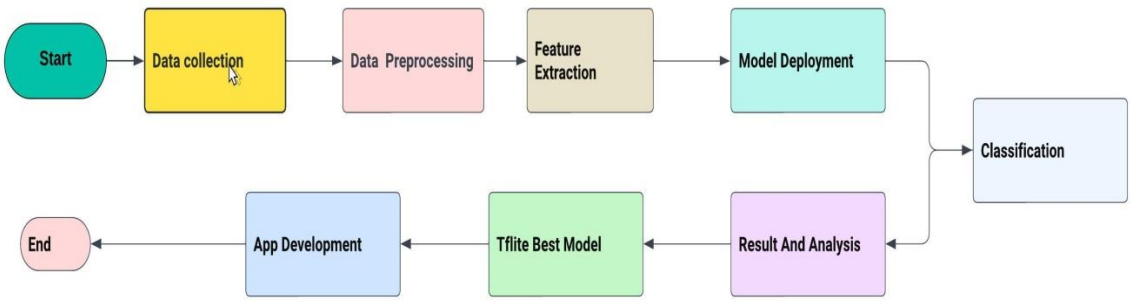


Figure 3.2: This is a sample data Flow diagram

3.1.6 UI Design

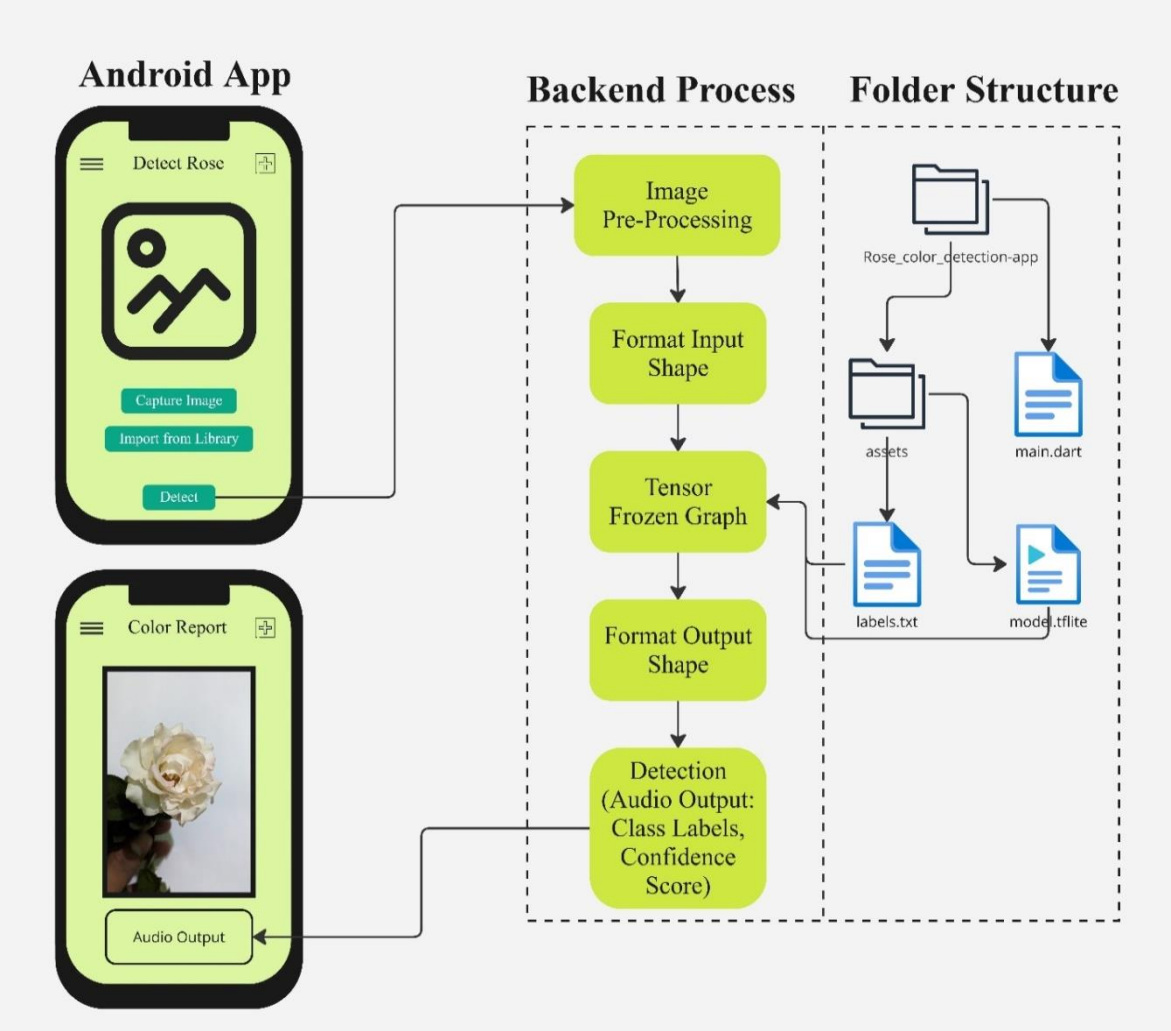


Figure 3.3: This is a sample of UI

3.2 Detailed Methodology and Design

The dataset used in this study is a collection of high-quality rose images. It includes a total of 886 original images, each captured at a high resolution of 3024x4032 pixels and saved in .jpg format. This high resolution allows the model to capture fine details and subtle color differences in the roses, which is important for accurate color detection.

Before feeding the images into the model, several preprocessing steps were applied to ensure all images are consistent and optimized for training. These steps included normalization, which adjusts pixel values to a standard range, making it easier for the model to learn patterns. The DPI (dots per inch) was also adjusted to maintain image quality while reducing processing requirements. Additionally, all images were resized to a consistent size that fits the input requirements of the model, allowing it to process the images efficiently.

Yellow_Rose White_Rose Orange_Rose Red_Rose

Figure 3.4: Sample Data for Each Class

Since the original dataset alone might not be enough for training a robust model, data augmentation techniques were applied during model training to increase the dataset's diversity. Data augmentation helps the model learn to recognize roses in different orientations, sizes, and lighting conditions. Through augmentation, the dataset was expanded to a total of 5316 images. These augmented images were divided into three sets: 3922 images were used for training the model, 862 images were used for validation (to tune and check the model during training), and 532 images were reserved for testing, ensuring the model's accuracy on unseen data.

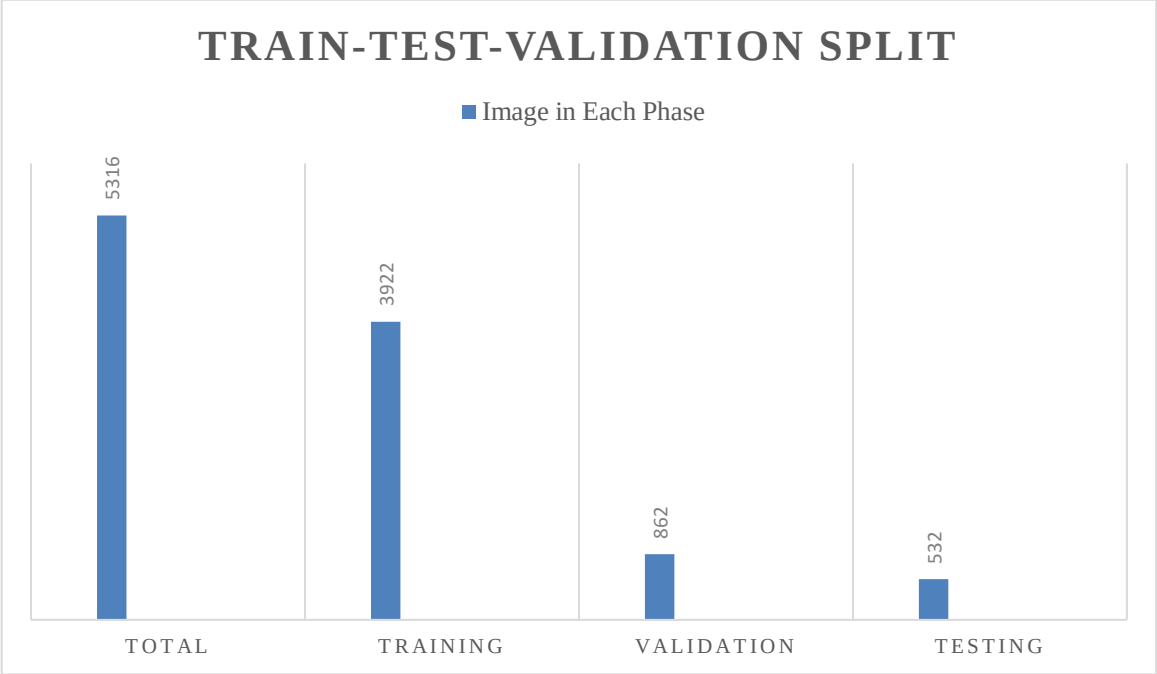


Figure 3.5: Train-Test-Validation Split visualization

The dataset is divided into four classes, each representing a different color of rose. The four classes are Yellow_Rose, White_Rose, Orange_Rose, and Red_Rose. This classification helps the model learn to differentiate between various colors, making it

easier to detect the specific rose color in real-world applications. By training on these distinct color classes, the model will be able to assist visually impaired users in accurately identifying rose colors through an accessible application.

Table 3.1: Dataset Specification

Properties	Values
Image Resolution	3024 * 4032 pixels
Format	.jpg
Total Images (Original)	886
Total Images (Augmented)	5316
Classes	4

3.2.1 Model Description

3.2.1.1 Vision Transformer (ViT)

The Vision Transformer (ViT) is a deep learning model that applies the Transformer architecture, originally developed for natural language processing, to image recognition tasks. Unlike convolutional neural networks (CNNs), which rely on convolutions to capture spatial patterns, ViT treats images as sequences of patches, allowing it to learn long-range dependencies and global contextual information effectively. Below is a step-by-step description of the ViT model process, from data augmentation and image preprocessing to the final output.

3.2.1.2 Data Augmentation

To enhance the model's robustness and generalization ability, data augmentation is applied during the training phase. Data augmentation techniques involve creating additional, modified versions of the original images by applying random transformations, helping the model generalize to new variations and reducing the risk of overfitting.

The following transformations are typically applied:

- **Rotation**: Randomly rotating images within a certain range to make the model invariant to orientation changes.
- **Flipping**: Horizontally flipping images to increase variety in the dataset.
- **Scaling**: Randomly zooming in or out, allowing the model to recognize roses of varying sizes.

By applying these transformations, the dataset is expanded from 886 original images to a

total of 5316 images, split into 3922 training images, 862 validation images, and 532 testing images.

3.2.1.3 Image Loading and Preprocessing

Each image in the augmented dataset is then loaded and preprocessed to fit the ViT model's input requirements. The preprocessing includes resizing, normalization, and adjusting the DPI.

- **Resizing and Normalization**

  Each image is resized to a consistent size, ensuring compatibility with the model's patch size requirements. Additionally, pixel values are normalized to a standard range (usually [0, 1] or [-1, 1]) to facilitate faster and more stable training.

- **Splitting Image into Patches**

  In the ViT model, each image is split into small, non-overlapping square patches of size P×P. Suppose each image has a height H, width W, and number of channels C; the number of patches N is calculated as:

$$N = \frac{H \times W}{P^2}$$

  Each patch is then flattened into a vector, creating a sequence that the Transformer model can process.

3.2.1.4 Patch Embedding

Once the image is split into patches, each patch is linearly embedded into a feature vector of a fixed size D. This embedding is achieved using a trainable linear projection, which converts each patch (originally a P×P×C vector) into a D-dimensional vector. The embedding process can be represented as:

$$z_0^p = x_p \cdot E$$

where:

- E is the learnable embedding matrix,
- z_0^p is the embedded patch.

This results in a sequence of N patch embeddings, each of dimension D.

3.2.1.5 Adding Position Embeddings

Transformers lack an inherent understanding of spatial relationships. To retain spatial order information, positional embeddings are added to each patch embedding. These positional embeddings are vectors of the same size as the patch embeddings, ensuring that each patch is aware of its position within the image. Let p_i represent the positional

embedding for the iii-th patch; the input embeddings z_0 are then defined as:

$$z_0 = [z_0^1 + p_1, z_0^2 + p_2, \ldots, z_0^N + p_N]$$

This sequence z_0 becomes the input to the Transformer encoder.

3.2.1.6 Transformer Encoder Layers

The Transformer encoder, the core of the Vision Transformer, processes the input sequence through multiple layers. Each layer consists of a multi-head self-attention mechanism and a feed-forward network, with layer normalization and residual connections applied at each stage.

- **Multi-Head Self-Attention**

The self-attention mechanism enables the model to capture dependencies between each pair of patches. For each patch embedding, the model computes three matrices — Queries Q, Keys K, and Values V — using learned weights:

$$Q = z \cdot W_Q, \quad K = z \cdot W_K, \quad V = z \cdot W_V$$

Where W_Q,W_K, and $W_V$are the weight matrices for queries, keys, and values, respectively.

The attention scores for each pair of queries and keys are computed as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where d_k is the dimensionality of the keys (used for scaling). The softmax function normalizes the scores, giving the attention weights for each patch.

- **Feed-Forward Network (FFN)**

After the attention mechanism, each patch embedding passes through a feed-forward network, consisting of two fully connected layers with a ReLU activation function:

$$\text{FFN}(x) = \text{ReLU}(x \cdot W_1 + b_1) \cdot W_2 + b_2$$

where W_1, b_1, W_2 and b_2 are trainable parameters. The FFN helps the model capture complex patterns and dependencies within the image.

Each layer output is normalized and combined with residual connections:

$$z_l' = \text{LayerNorm}(z_{l-1} + \text{MultiHeadAttention}(z_{l-1}))$$

$$z_l = \text{LayerNorm}(z_l' + \text{FFN}(z_l'))$$

This process is repeated across L Transformer layers.

3.2.1.7 Classification Token and Output

A special *class token* is preceded to the sequence of patches. This token gathers information from all patches through the self-attention layers, eventually representing the entire image. After the Transformer layers, the output of the class token is passed through a fully connected layer with softmax activation, producing probabilities for each rose color class.

Let $z_{[CLS]}$ be the output of the class-token after the encoder layers. The final class probabilities are calculated as:

$$\hat{y} = \text{softmax}(z_{[CLS]} \cdot W_C + b_C)$$

where W_C and b_C are the weights and biases of the classification layer.

3.2.1.8 Model Output

The final output stage of the Vision Transformer (ViT) model is where the actual classification decision is made. This step involves interpreting the features learned from the image and generating a probability distribution over the predefined classes of rose colors. In this study, the model is designed to classify images into four distinct rose color categories: Yellow_Rose, White_Rose, Orange_Rose, and Red_Rose.

- **Generating Class Probabilities**

  After passing through multiple layers of the Transformer encoder, including the multi-head self-attention and feed-forward network layers, the *class token* has accumulated information from all patches of the image. This class token, denoted by $z_{[CLS]}$ represents a holistic understanding of the entire image and serves as the final embedding that encapsulates all relevant features for classification.

  The model then takes this final embedding, $z_{[CLS]}$ and feeds it through a fully connected layer, often referred to as the *classification head*. This layer projects the embedding into a vector with dimensions equal to the number of classes, effectively mapping the learned features to specific categories. Mathematically, this process can be represented as:

$$\hat{y} = z_{[CLS]} \cdot W_C + b_C$$

  where:

  - Wc is the weight matrix of the classification layer, and
  - b_c is the bias vector.

The resulting vector, $\hat{y}$ contains raw scores (also known as *logits*) for each class.

These logits are not directly interpretable as probabilities, so they are passed through a softmax activation function to transform them into a probability distribution.

- **Applying Softmax Activation**

The softmax function converts the raw scores into probabilities, ensuring that the sum of the probabilities across all classes equals 1. The probability for each class i is given by:

$$P(\text{Class}_i) = \frac{\exp(\hat{y}_i)}{\sum_{j=1}^{C} \exp(\hat{y}_j)}$$

where:

- is the raw score for class iii,
- C is the total number of classes (in this case, 4), and
- exp (·) denotes the exponential function.

This transformation maps each score into a value between 0 and 1, making it interpretable as the probability that the image belongs to a specific class. The class with the highest probability is selected as the final prediction of the model.

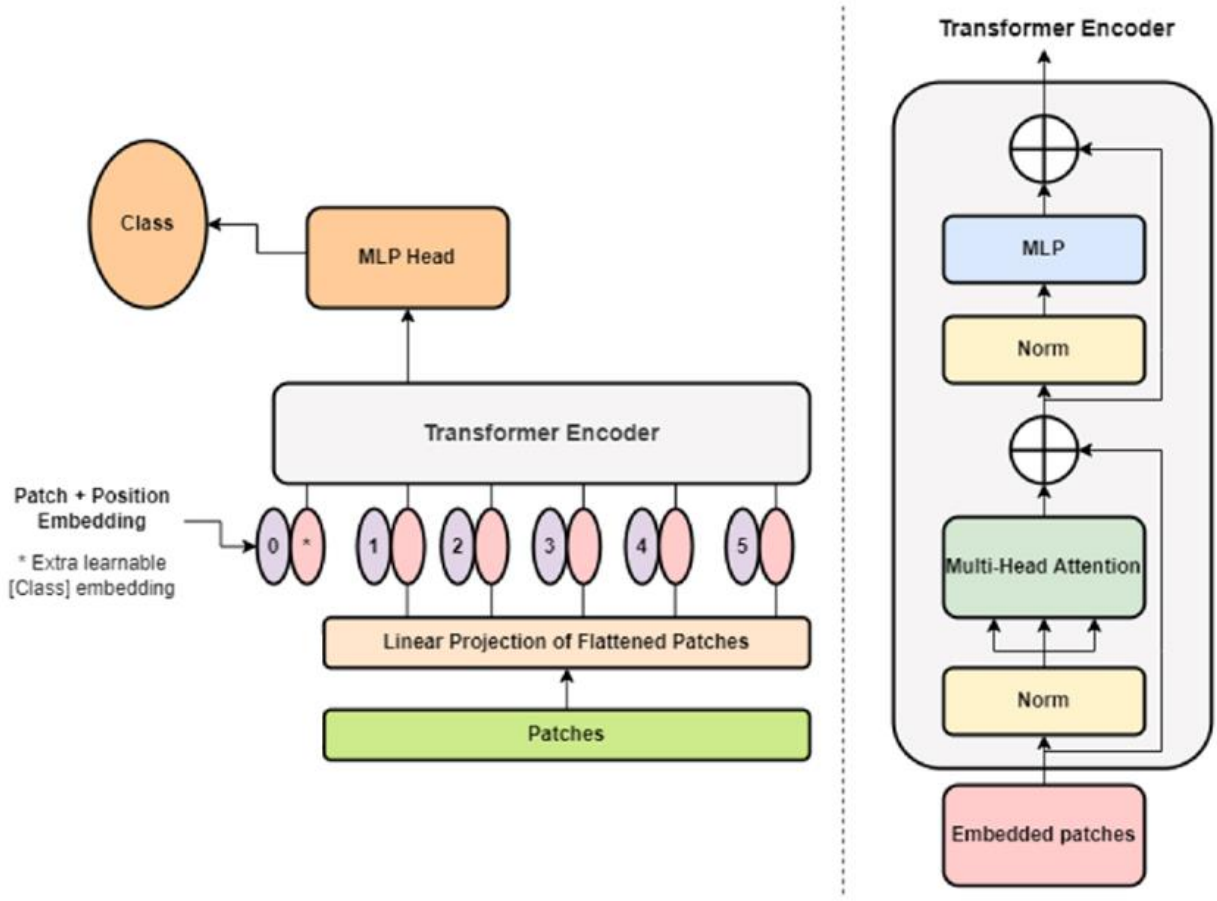


Figure 3.6: The Architecture of Pre-Trained Vision Transformer (ViT) [1]

3.2.2.1 Comparison of Pre-trained Vision Transformers (ViTs)

In this study, four configurations of the Pre-trained Vision Transformer (ViT) model were evaluated to determine the most suitable model for rose color classification. These configurations vary by image size and patch size, influencing how each model captures spatial and color information from the images. The four specific configurations tested are:

1. ViT-b16 with an image size of 32x32
2. ViT-b16 with an image size of 64x64
3. ViT-b32 with an image size of 32x32
4. ViT-b32 with an image size of 64x64

Each configuration has unique advantages and trade-offs based on the model's ability to balance computational efficiency and accuracy in classification. Below is a detailed comparison of these configurations across key parameters.

3.2.2.2 Key Parameters for Comparison

1. **Image Size**: The input image size significantly impacts the amount of detail the model can capture. Unlike (32 x 32) higher resolutions (64 x 64) offer more visual information but require greater computational resources.
2. **Patch Size**: The ViT-b16 and ViT-b32 configurations refer to the patch sizes. ViT-b16 splits each image into smaller patches, allowing the model to capture finer details, which is beneficial for capturing subtle color variations in roses. In contrast, ViT-b32 uses larger patches, which may reduce computational cost but at the potential expense of fine-grained feature capture.
3. **Feature Extraction Capability**: Models with smaller patch sizes (b16) generally have higher feature extraction capability due to their ability to capture smaller details. This characteristic can be particularly useful for tasks requiring precise color and texture differentiation, such as rose color classification.
4. **Computational Efficiency**: Larger patch sizes (b32) and smaller image sizes (32x32) lead to faster computations and lower memory usage, making them suitable for real-time applications on mobile or low-power devices. However, this comes at a potential trade-off in accuracy.
5. **Accuracy and Robustness**: Smaller patch sizes and higher image resolutions often lead to higher accuracy, as these configurations capture more details. However, the increased accuracy may come with higher computational demands, which can be a limitation for deployment on devices with limited processing power.

Table 3.2: Proposed Table for Comparison

Configuration	Image Size	Patch Size	Feature Extraction Capability	Computational Efficiency	Accuracy and Robustness
ViT-b16 (32x32)	32x32	16x16	Moderate	High	High
ViT-b16 (64x64)	64x64	16x16	High	Moderate	Moderate
ViT-b32 (32x32)	32x32	32x32	Low	Very High	Moderate
ViT-b32 (64x64)	64x64	32x32	Moderate	Moderate	Moderate

Analysis of Configurations

1. **ViT-b16 (32x32)**: This configuration with an image size of 32x32 and patch size of 16x16 offers a balanced approach, providing moderate feature extraction and high computational efficiency. This model captures sufficient detail for general rose color classification without overwhelming computational resources. However, it may struggle with finer details compared to higher resolution models.

2. **ViT-b16 (64x64)**: The 64x64 image size and 16x16 patch size enable this configuration to capture fine-grained details. It offers high feature extraction capability, which is valuable for capturing subtle color variations in roses. While it requires more computational power than the 32x32 configurations, it provides higher accuracy and robustness, making it ideal for applications prioritizing precision.

3. **ViT-b32 (32x32)**: With an image size of 32x32 and patch size of 32x32, this configuration focuses on computational efficiency, making it the fastest among the four. However, its large patch size limits its ability to capture small details, which may affect accuracy in tasks requiring fine color distinctions.

4. **ViT-b32 (64x64)**: This model strikes a balance between accuracy and efficiency. With a larger image size (64x64) but a bigger patch size (32x32), it achieves moderate feature extraction, capability and accuracy. This configuration may be

suitable for applications where moderate detail is needed without overwhelming computational resources.

3.2.2.3 Application Setup

The rose color detection application is designed to assist visually impaired individuals in identifying rose colors by utilizing machine learning and deep learning techniques. The application's primary setup is structured to ensure seamless integration of the Vision Transformer (ViT) model for image classification, as well as a user-friendly interface and audio feedback for accessibility.

The application is developed using the Flutter framework, which enables cross-platform functionality, allowing it to run on both Android and iOS devices. This ensures that a broad range of users can benefit from the application, regardless of the operating system on their device. Flutter's native-like performance, fast development cycle, and customizable widgets make it an ideal choice for building an accessible and interactive application.

The application workflow begins with the user capturing or selecting an image of a rose. The image is then preprocessed and passed through the ViT model, which classifies the rose into one of four color categories: Yellow, White, Orange, or Red. After classification, the result is displayed on the screen and conveyed through audio output, enabling visually impaired users to receive the information audibly.

The application is built using a Two-Tier Architecture, which effectively separates the machine learning model processing (backend) from the user interaction interface (frontend). This architecture provides a clear separation of concerns, improves maintainability, and ensures that each tier can be modified independently if needed.

3.2.2.4 Two-Tier Architecture

The two-tier architecture divides the application into two main layers: the Presentation Layer (Frontend) and the Application Logic Layer (Backend). Each layer has specific responsibilities that contribute to the overall functionality of the application.

3.2.2.5 Presentation Layer (Frontend)

The Presentation Layer is the user-facing part of the application, built using Flutter. This layer is responsible for all interactions between the application and the end user. It includes the user interface (UI), which consists of components such as buttons, image display areas, and audio feedback controls. This layer ensures that users can interact with the application in a simple and intuitive way.

Key components of the Presentation Layer include:

- **User Interface (UI)**: The UI is designed to be accessible for visually impaired users. The main screen includes a "Detect" button, allowing users to trigger the rose color detection process. After classification, the UI displays the detected color name visually.

- **Audio Output**: For accessibility, the application includes text-to-speech (TTS) functionality. Once the rose color is detected, the application uses TTS to verbally announce the color. This feature is essential for visually impaired users who rely on audio cues rather than visual output.

- **Image Input**: Users can capture or select an image of a rose through the app's interface. This image is then sent to the backend for processing and classification.

By isolating the presentation layer, the application can ensure a responsive and smooth user experience, allowing modifications to the UI without affecting the backend logic.

3.2.2.6 Application Logic Layer (Backend)

The **Application Logic Layer** handles the core functionality of the application, including image processing, model inference, and data management. This layer is responsible for loading and executing the Vision Transformer model, which performs the actual classification of the rose color. It operates independently of the frontend, focusing solely on processing inputs and delivering results back to the presentation layer.

Key components of the Application Logic Layer include:

- **Model Inference**: The backend loads the pre-trained Vision Transformer model, which has been converted to a TensorFlow Lite (.tflite) format for optimized performance on mobile devices. This model processes the input image, extracting relevant features and generating a classification prediction. The model inference is designed to be lightweight, allowing fast processing even on devices with limited computational power.

- **Data Preprocessing**: Before sending the image to the model, the backend preprocesses it by resizing, normalizing, and adjusting the DPI. These steps ensure that the image matches the model's input requirements, leading to accurate classification results.

- **Classification Logic**: The backend applies to the ViT model to classify the image into one of four rose color categories. The model returns a probability distribution across the classes, and the class with the highest probability is selected as the detected color. This result is then sent back to the presentation layer.

- **Audio Integration**: Once the classification result is determined, the backend sends a signal to the front end to initiate audio output. The result is converted into a text format that the TTS system reads aloud, providing an audio description of the detected color.

By separating the application logic from the presentation layer, the two-tier architecture enhances maintainability and scalability. The backend can be modified independently (by updating the model or improving pre-processing steps) without requiring changes to the front end. Additionally, this architecture supports efficient communication between layers, ensuring that classification results are quickly returned to the user interface.

3.3 Project Plan

Table 3.3: GANTT Chart of Project Timeline

Process	May'24	June'24	July'24	Aug'24	Sep'24	Oct'24	Nov'24	Dec'24
Working Plan								
Theoretical Study								
Literature Review								
Primary Data Collection								
Model Design								
Application Development								
Methodology Writing								
Report Writing								
Review and Finalization								

3.4 Task Allocation

This team is solo. So, I handled the total project alone.

Task	Team Member (Sajeeb Mandal)
Choose Rose	Yes
Data Collection	Yes
Data Preprocessing	Yes
Model Selection	Yes
Model Run	Yes
Report Writing	Yes

Table 3.4: Task Allocation table.

3.5 Summary

The methodology for this thesis focuses on developing a rose color detection application for visually impaired individuals, utilizing machine learning and deep learning techniques. The project began with data collection and preprocessing, where a dataset of high-resolution rose images was compiled. To enhance model accuracy and robustness, data augmentation techniques were applied, including rotation, flipping, and scaling, resulting in an expanded dataset suitable for training. Each image was resized, normalized, and split into patches, following the requirements for the Vision Transformer (ViT) model. The dataset was organized into four classes: Yellow_Rose, White_Rose, Orange_Rose, and Red_Rose, allowing the model to effectively classify rose colors.

The core of the model development involved experimenting with multiple configurations of the ViT architecture, including variations in image and patch sizes. These configurations, such as ViT-b16 with 32x32 or 64x64 image sizes, were compared based on feature extraction capability, computational efficiency, and accuracy. After evaluating these variations, the optimal configuration was selected, and the trained model was converted into a TensorFlow Lite format (.tflite) to enable efficient inference on mobile devices. This model was integrated into a Flutter-based mobile application, designed to run on both Android and iOS platforms, allowing the application to be cross-platform and accessible to a broader audience.

The application was developed with a two-tier architecture, separating the presentation

layer (frontend) from the application logic layer (backend). The frontend, created with Flutter, included a user-friendly interface with large buttons and text-to-speech functionality for accessible interaction. The backend handled model inference and data processing, ensuring accurate classification and quick response times. The classification results, including the detected rose color, were displayed visually and conveyed through audio output to aid visually impaired users.

In managing this project, a clear strategy was established to address potential risks and maintain budget efficiency. Risk management focused on mitigating technical, data, and financial risks. Technical risks, such as model underperformance or compatibility issues, were addressed through extensive model testing and optimization. Data risks, including quality and storage limitations, were mitigated by using high-resolution images and implementing data augmentation techniques.

Financial management was carefully planned to keep costs low. Google Colab provides free or low-cost cloud resources for model training, while open-source software, such as TensorFlow and Flutter, minimized software expenses. Overall, this approach ensured that project risks were managed effectively, and resources were used efficiently, supporting the development of an accessible and cost-effective solution for rose color detection.

Chapter 4

Implementation and Results

4.1 Environment Setup

This table outlines the key parameters used to train the model. The image size is set to 32 × 32, indicating that each input image is resized to this resolution before being processed by the model. This compact image size helps reduce computational requirements and enables the model to process multiple images efficiently. The batch size is set to 32, meaning that 32 images are processed at a time before the model updates its weights. This batch size is a balance between computational efficiency and stable learning, helping to achieve more accurate gradient estimates while conserving memory.

Table 4.1: Common parameter table for all experimented models.

Parameter Name	Parameter Value
Image Size	32 × 32
Batch Size	32
Epoch	15
Optimizer	Adam
Learning Rate	1e - 4

The model is trained for 15 epochs, meaning the entire dataset is passed through the model 15 times during training. This number of epochs is moderate and suggests that the training process aims for quick convergence while minimizing the risk of overfitting. The optimizer used is Adam (Adaptive Moment Estimation), a popular and efficient

optimization algorithm that adapts the learning rate for each parameter. Adam combines the benefits of the Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp), making it effective for handling sparse gradients and noisy data. The learning rate is set to 1e-4 (0.0001), which controls the step size for each iteration as the model converges to minimize the loss function. This relatively low learning rate allows the model to learn gradually, reducing the chance of overshooting the optimal solution and helping ensure a smoother convergence path. Overall, these parameters are chosen to enable efficient, stable, and accurate training of the model.

Table 4.2: Common data split for all experimented ViT models.

Dataset	In Percentage	Number of Images
Train set	72%	3922
Validation Set	18%	862
Test Set	10%	532

Table 2 describes the data split used for training, validating, and testing all Vision Transformer (ViT) models in this study, ensuring consistency across experiments. The train set comprises 72% of the dataset, with 3922 images, providing a substantial portion of data for the model to learn patterns and features effectively. This large training set helps the model generalize better by exposing it to diverse examples. The validation set includes 18% of the data, amounting to 862 images, and is used to monitor the model's performance during training. By evaluating the model on this unseen validation data, we gain insights into its generalization ability and can adjust hyperparameters as needed to prevent overfitting. Finally, the test set represents 10% of the dataset, with 532 images, and is reserved exclusively for final evaluation. This test set provides an unbiased assessment of the model's real-world performance, as it has not been used in training or validation. This structured data split, with 72% for training, 18% for validation, and 10% for testing, ensures a balanced approach to model development, enabling accurate evaluation of the ViT models' effectiveness.

4.1.1 Model Evaluation

In evaluating the effectiveness of machine learning models for the study, appropriate performance metrics must be used to provide insights into model accuracy, reliability, and generalization capabilities. The following metrics are commonly used in classification tasks, especially in the context of agricultural disease detection:

4.1.2 Accuracy

The proportion of correctly classified instances (both positive and negative) to the total instances. Accuracy gives a quick overview of model performance but can be misleading in imbalanced datasets where one class significantly outnumbers the other.

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

4.1.3 Recall

The ratio of correctly predicted positive observations to all actual positives. Recall is particularly important in scenarios where a positive case (such as a diseased plant) could lead to severe consequences, like crop loss.

$$\text{Recall} = \frac{\text{TP}}{(\text{FN} + \text{TP})}$$

4.1.4 Precision

The ratio of correctly predicted positive observations to the total predicted positives. Precision is crucial in applications where the cost of false positives is high. In this study, high precision indicates that when a disease is predicted, it is likely to be true.

$$\text{Precision} = \frac{\text{TP}}{(\text{FP} + \text{TP})}$$

4.1.5 F1-Score

The harmonic meaning of precision and recall, provides a balance between the two metrics. The F1 score is especially useful when dealing with imbalanced classes, as it considers both false positives and false negatives, offering a more comprehensive view of model performance.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} + \text{Recall}}{(\text{Precision} \times \text{Recall})}$$

4.1.6 Confusion Matrix

A table used to describe the performance of a classification model, showing the true vs. predicted classifications. The confusion matrix provides insights into the types of errors made by the model, allowing for more targeted improvements.

Receiver Operating Characteristic (ROC) Curve

A graphical representation of a classifier's performance across various threshold settings, plotting the true positive rate (recall) against the false positive rate. It illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. It plots the True Positive Rate (TPR), also known as recall or sensitivity, against the False

Positive Rate (FPR) at various threshold settings.

Area Under Curve (AUC): The area under the ROC curve, provides a single metric to assess model performance; a value closer to 1 indicates a better model.

In this study, the Receiver Operating Characteristic (ROC) curve and Area Under the Curve (AUC) are crucial for evaluating classification models. They allow for visual comparisons of model performance across various thresholds, facilitating optimal threshold selection by balancing sensitivity and specificity. Additionally, the ROC curve is robust against class imbalances, providing reliable assessments where accuracy may mislead. However, ROC and AUC should complement other metrics like precision and recall for a comprehensive evaluation of model effectiveness.

4.2 Testing and Evaluation

In summary from subsection 5.2, the ViT B16 model with a 32×32 image size achieves the most robust performance across all metrics, with perfect or near-perfect accuracy, AUC scores, and consistent class separation. Both ViT B16 and B32 models trained on a 64×64 image size performs slightly lower due to minor misclassifications, particularly between visually similar classes such as white rose and yellow rose. Overall, while increasing image resolution can add detail, it may also introduce complexity, impacting model stability. Based on these findings, the ViT B16 (32×32) configuration is recommended for tasks requiring high accuracy and reliable generalization across classes.

Table 4.3: Performance analysis among all the experimented ViT models and image sizes.

Model Name	Image Size	Validation Accuracy	Test Accuracy	Average AUC score
B16	32 × 32	99 %	99.81 %	1.00
B16	64 × 64	100 %	99.44 %	1.00
B32	32 × 32	100 %	99.62 %	1.00
B32	64 × 64	99 %	99.44 %	1.00

The performance results show that all four configurations achieve high validation and test accuracies, along with perfect AUC scores, but with slight variations. The ViT B32

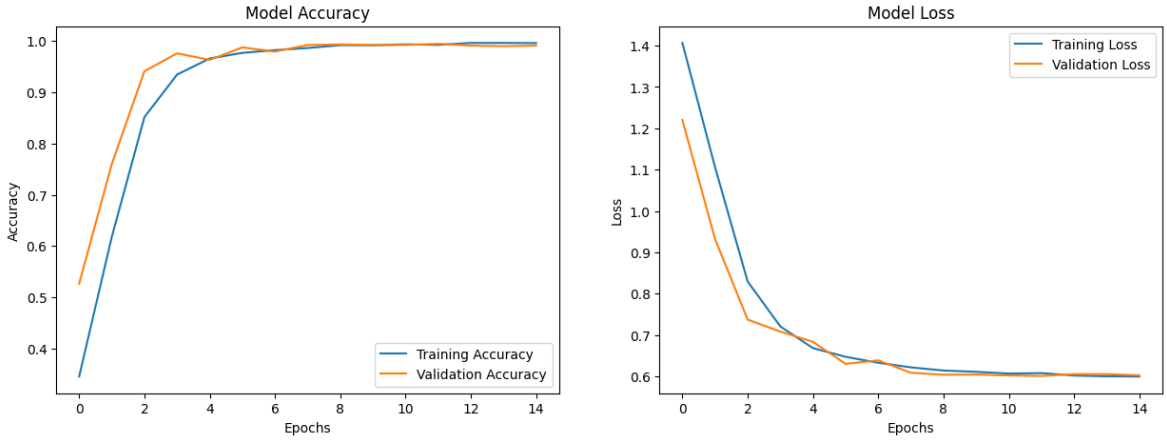
model with a 64 × 64 image size reaches the highest validation accuracy of 100% and a test accuracy of 99.44%, demonstrating exceptional learning capability and strong generalization. Meanwhile, the ViT B16 model with a 32 × 32 image size achieves 99% and 100% validation accuracy in two configurations, with test accuracies of 99.81% and 99.62%, indicating consistent performance and minimal overfitting. Both ViT B32 configurations with a 64 × 64 image size have a slight drop in test accuracy to 99.44% despite their high validation accuracy, which may reflect sensitivity to unseen data variations. All models attain an AUC of 1.00, indicating perfect class separability and strong discriminative power. While all configurations perform exceptionally well, the ViT B16 (32 × 32) model with a test accuracy of 99.81% and stable validation accuracy is the most reliable choice, balancing both accuracy and consistency across validation and test sets.

4.3 Results and Discussion

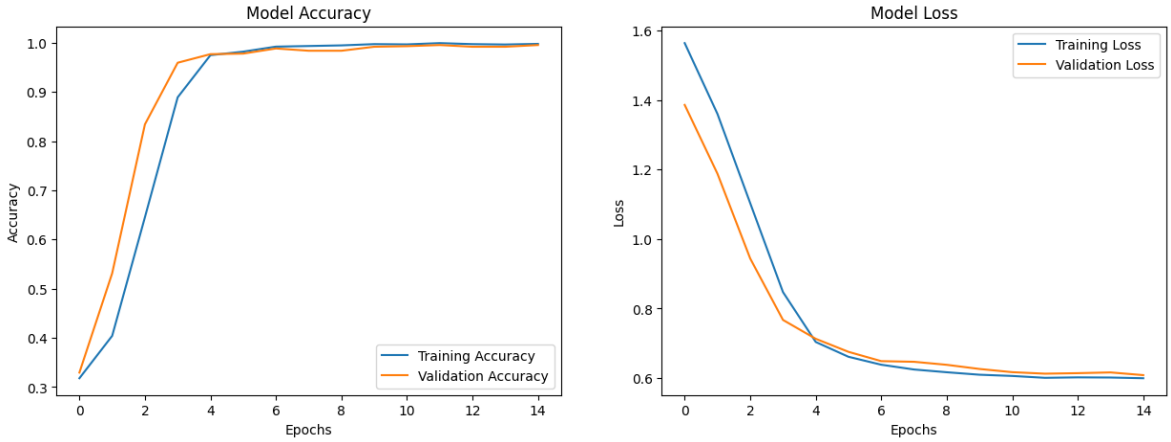
This analysis explores four Pretrained ViT models: ViT B16 and ViT B32, each evaluated with image sizes of 32×32 and 64×64. Each model's performance is assessed based on loss and accuracy curves, confusion matrices, classification reports, and ROC curves to understand how the model type and image resolution impact performance on the classification task.

The loss and accuracy curves over 15 epochs reveal the learning behavior and stability of each ViT model configuration. The ViT B16 model with a 32×32 image size exhibits a steady reduction in both training and validation loss, reaching high accuracy by the end of the epochs. It achieves near-perfect validation accuracy, indicating strong generalization with minimal overfitting. Similarly, the ViT B32 model with a 32×32 image size follows a comparable trend, achieving stable convergence and high accuracy, although it shows slight fluctuations, suggesting mild sensitivity to the training data. When increasing the image size to 64×64, both ViT B16 and B32 models display similar trends but with minor increases in validation loss fluctuation. This could indicate that while larger image sizes provide more detail, they introduce additional complexity, which may impact model consistency. Overall, both configurations and image sizes achieve high accuracy, but ViT B16 with 32×32 image size shows the most stable performance in terms of accuracy and loss balance.

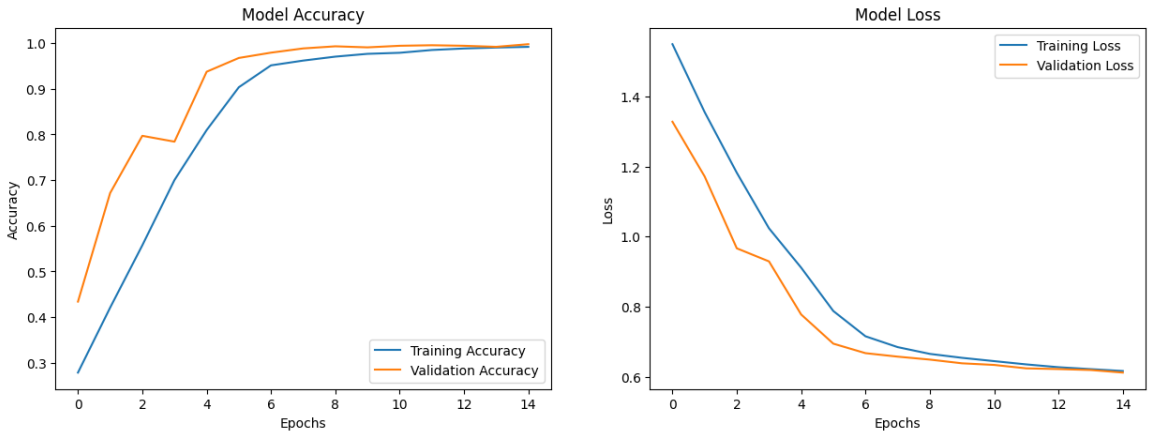
ViT B16 (Image size – 32 × 32)



ViT B16 (Image size – 64 × 64)



ViT B32 (Image size – 32 × 32)



ViT B32 (Image size – 64 × 64)

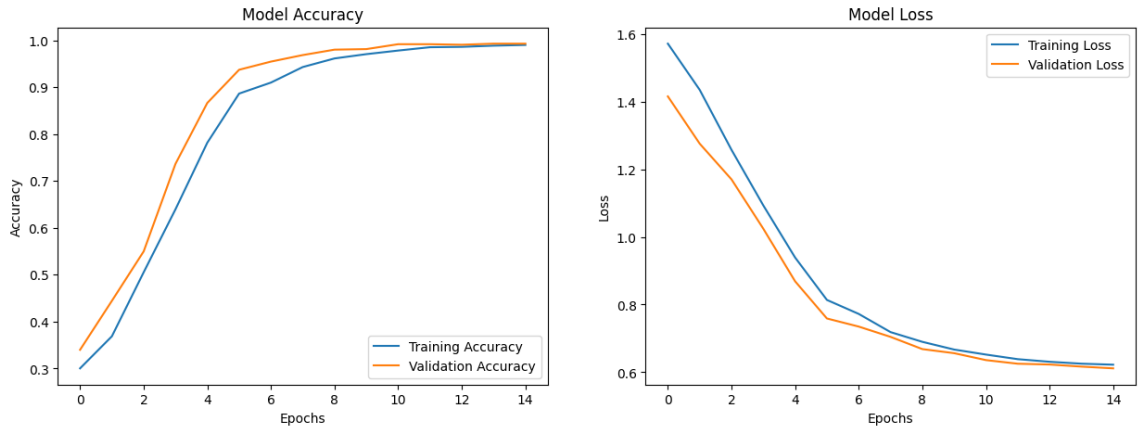
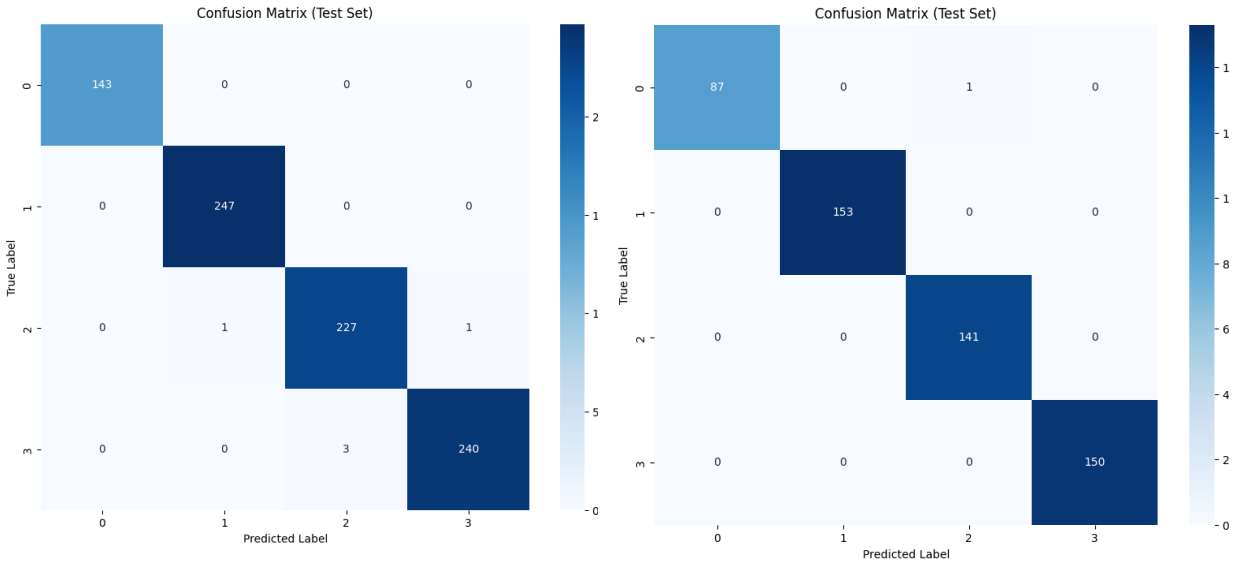


Figure 4.1: The loss and accuracy curve of different Pretrained ViT models and image sizes over epoch 15.

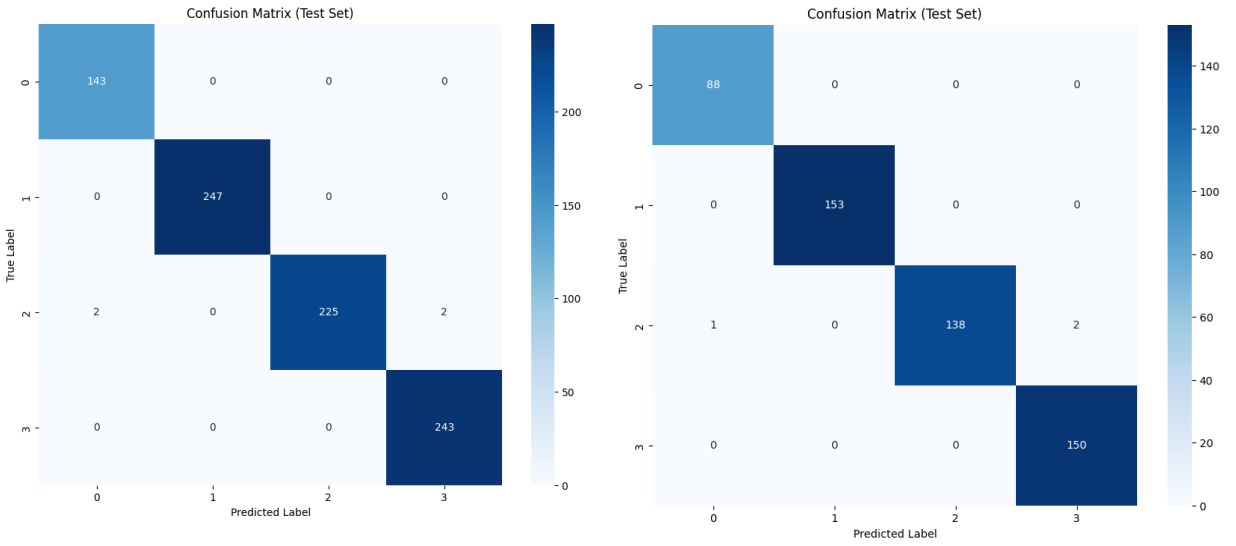
The validation and test confusion matrices for each model reveal their classification accuracy across four classes: orange rose, red rose, white rose, and yellow rose. The ViT B16 model (32×32) achieves nearly perfect classification, with minimal to no misclassifications across all classes on both the validation and test sets. This indicates strong feature extraction and robust classification capabilities for this configuration. Similarly, the ViT B32 model (32×32) performs almost flawlessly, though it occasionally misclassifies instances between white rose and yellow rose, likely due to overlapping features. For models trained with a 64×64 image size, both ViT B16 and B32 maintain high accuracy but show slight increases in misclassification, particularly in the white rose and yellow rose classes on the test set. This suggests that while higher-resolution images provide finer details, they may introduce complexities that slightly challenge the model's ability to distinguish similar classes.

Validation Testing

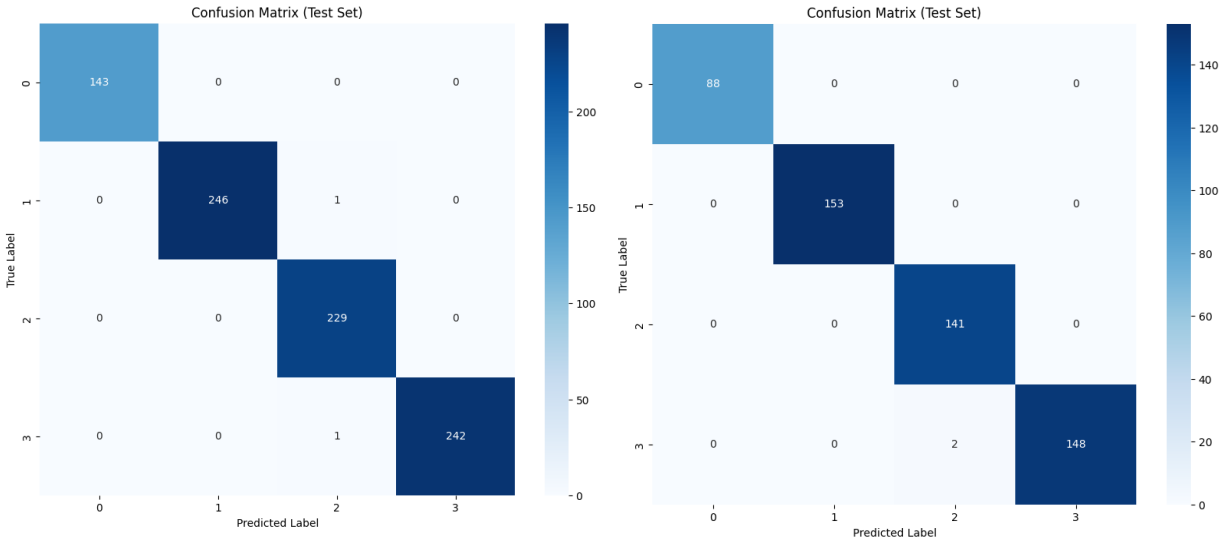
ViT B16 (Image size – 32 × 32)



ViT B16 (Image size – 64 × 64)



ViT B32 (Image size – 32 × 32)



ViT B32 (Image size – 64 × 64)

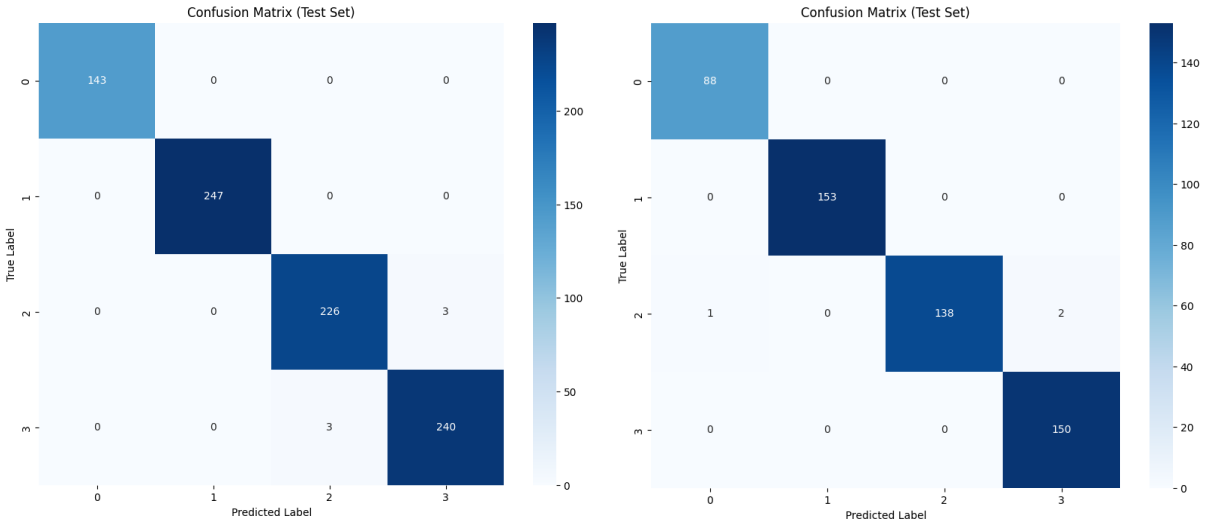


Figure 4.2: The validation and test confusion matrix of different Pretrained ViT models and image sizes.

The classification report provides detailed metrics for each model's performance on the test set, including precision, recall, and F1-score for each class. The ViT B16 (32×32) configuration achieves perfect scores across all metrics, with precision, recall, and F1-score of 1.00 for each class, resulting in an overall accuracy of 100%. This shows the model's exceptional ability to avoid both false positives and false negatives, making it the most effective in terms of accurate classification. The ViT B16 (64×64) model has a slight decrease in scores, with precision and F1-scores of 0.99 for some classes,

especially for a white rose, resulting in an overall accuracy of 99%. Similarly, the ViT B32 (32×32) model achieves near-perfect scores, with minor drops in recall for yellow rose, maintaining an overall accuracy of 99%. Lastly, the ViT B32 (64×64) model shows similar minor drops in precision and recall for specific classes, particularly white rose, with an overall accuracy of 99%. These results highlight that while all configurations perform exceptionally well, ViT B16 with 32×32 image size remains the most accurate.

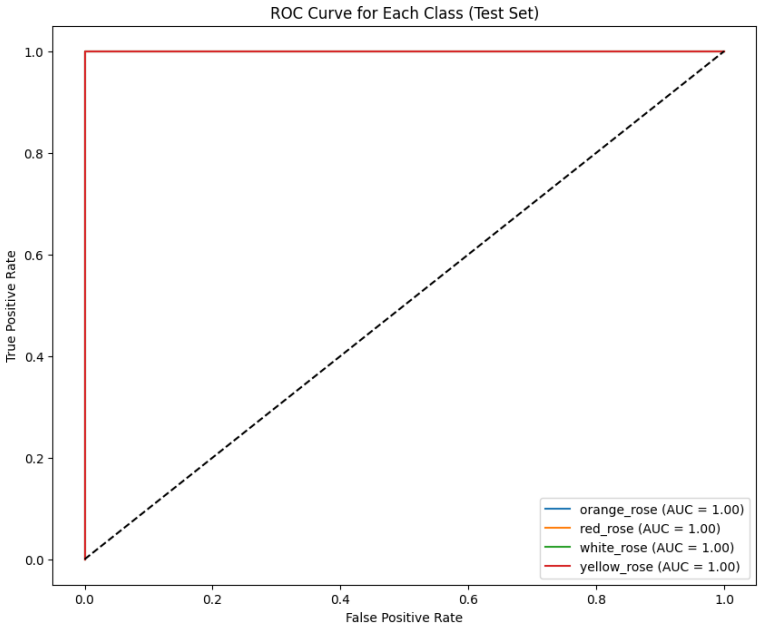
Table 4.4: The classification report on the test set of different Pretrained ViT models and image sizes.

Classes	Precision	Recall	F1-score	Support
ViT B16 (Image size – 32 × 32)				
orange_rose	1.00	0.99	0.99	88
red_rose	1.00	1.00	1.00	153
white_rose	0.99	1.00	1.00	141
yellow_rose	1.00	1.00	1.00	150
accuracy			1.00	532
macro avg	1.00	1.00	1.00	532
weighted avg	1.00	1.00	1.00	532
ViT B16 (Image size – 64 × 64)				
orange_rose	0.99	1.00	0.99	88
red_rose	1.00	1.00	1.00	153
white_rose	1.00	0.98	0.99	141
yellow_rose	0.99	1.00	0.99	150
accuracy			0.99	532
macro avg	0.99	0.99	0.99	532
weighted avg	0.99	0.99	0.99	532
ViT B32 (Image size – 32 × 32)				
orange_rose	1.00	1.00	1.00	88

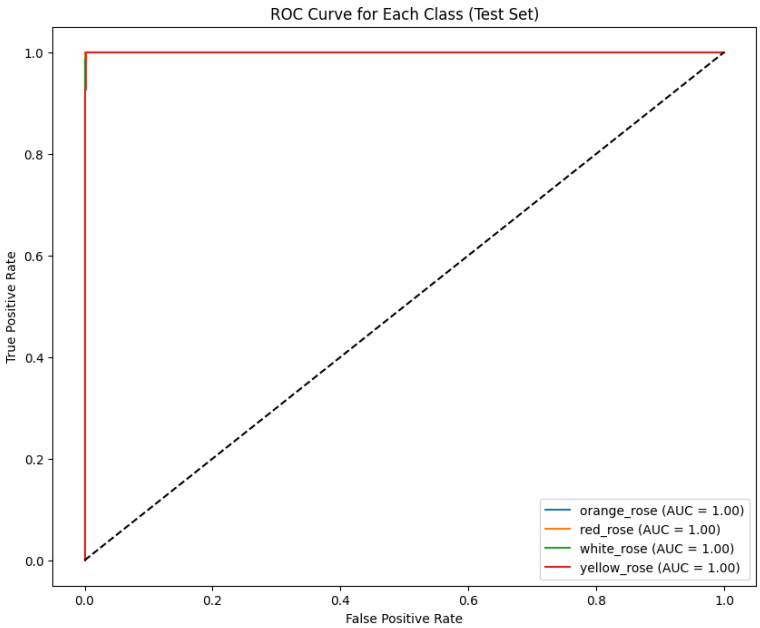
red_rose	1.00	1.00	1.00	153
white_rose	0.99	1.00	0.99	141
yellow_rose	1.00	0.99	0.99	150
accuracy			1.00	532
macro avg	1.00	1.00	1.00	532
weighted avg	1.00	1.00	1.00	532
ViT B32 (Image size – 64 × 64)				
orange_rose	0.99	1.00	0.99	88
red_rose	1.00	1.00	1.00	153
white_rose	1.00	0.98	0.99	153
yellow_rose	0.99	1.00	0.99	150
accuracy			0.99	532
macro avg	0.99	0.99	0.99	532
weighted avg	0.99	0.99	0.99	532

The ROC curves and AUC scores further illustrate the discriminative ability of each model configuration across classes. All models achieve nearly perfect AUC scores for each class, with the ViT B16 (32×32) model reaching an AUC of 1.00 for each class, reflecting perfect class separation. The ViT B32 (32×32) configuration also achieves high AUC scores, close to 1.00 for most classes, though with slight variations. Both models trained on a 64×64 image size (ViT B16 and B32) exhibit similarly high AUC scores, around 0.99, with slight drops in the white rose and yellow rose classes. These results indicate that all models are highly effective at distinguishing between classes, although ViT B16 with a 32×32 image size demonstrates the most consistent and precise class separation.

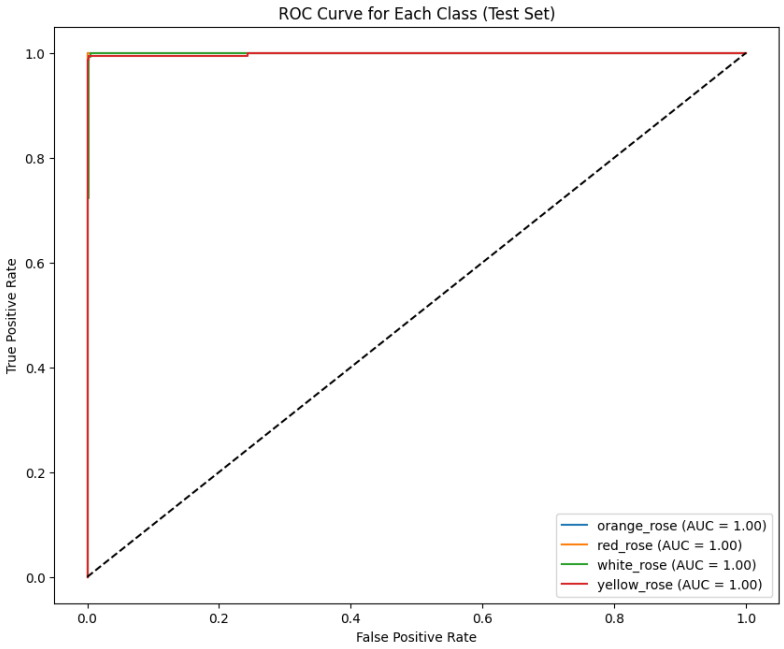
ViT B16 (Image size – 32 × 32)



ViT B16 (Image size – 64 × 64)



ViT B32 (Image size – 32 × 32)



ViT B32 (Image size – 64 × 64)

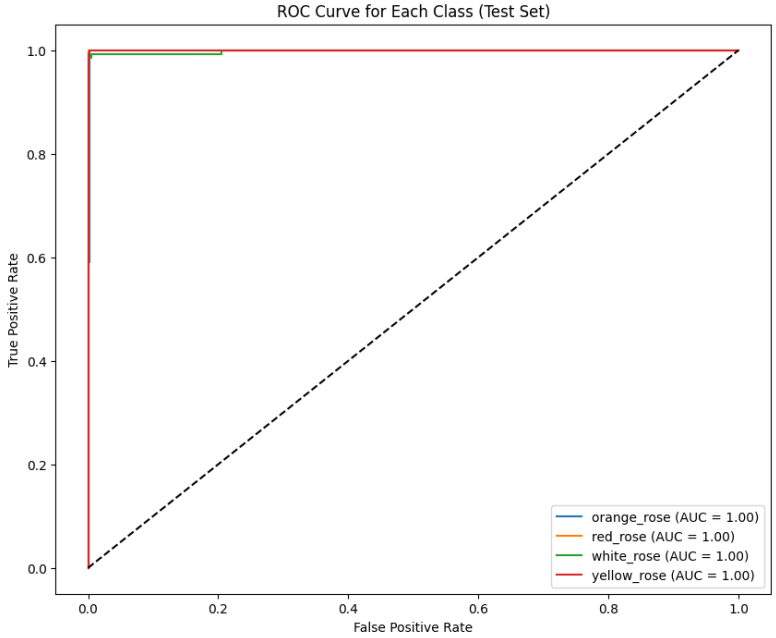


Figure 4.3: The ROC curve and AUC score for each class of different Pretrained ViT models and image sizes.

4.4 Summary

In summary, the ViT B16 model with a 32 × 32 image size emerges as the most effective configuration, achieving near-perfect accuracy and AUC scores while maintaining consistent performance across validation and test sets. The analysis reveals that while increasing the image resolution to 64 × 64 provides finer details, it may introduce complexities that slightly affect model stability, as seen in both ViT B16 and B32 models trained on this higher resolution. Across all configurations, the models achieve high validation and test accuracies, with perfect or near-perfect class separability as indicated by AUC scores. However, the ViT B16 (32 × 32) configuration stands out with the most balanced performance, combining high accuracy, strong generalization, and minimal misclassification, making it well-suited for reliable, real-world applications in image classification.

Chapter 5

Engineering Standards and Design Challenges

5.1 Compliance with the Standards

5.1.1 Software Standards

5.1.1.1 Software Requirements

The software requirements span across different stages of the experiment, from model training and preprocessing to mobile app development.

5.1.1.2 Model Training and Development

The following software tools and libraries were essential for training the Vision Transformer model, performing data preprocessing, and managing experiment workflows.

- **Google Colab**: A cloud-based environment with free access to GPUs for training and experimentation.
- **Python 3.x**: The programming language used for model development, data preprocessing, and experimentation.
- **TensorFlow and TensorFlow Lite**: TensorFlow was used to train the Vision Transformer model, while TensorFlow Lite was utilized to convert the trained model into a mobile-friendly format (.tflite) for efficient inference on mobile devices.

- **Transformers Library (Hugging Face)**: Provides pre-trained Vision Transformer (ViT) models and utilities for loading, fine-tuning, and experimenting with transformer architectures.
- **OpenCV**: Used for image preprocessing tasks such as resizing, normalization, and augmentation.
- **NumPy and Pandas**: Essential for data manipulation, augmentation, and handling the dataset during preprocessing.
- **Matplotlib**: For data visualization during the training phase, helping analyze model performance metrics.

5.1.2 Hardware Standards

5.1.2.1 Hardware Requirements

Since this experiment involves both model training and mobile application development, the hardware requirements can be divided based on these tasks. The training of the Vision Transformer (ViT) model and data preprocessing were performed on Google Colab, while the application is intended to run on mobile devices.

5.1.2.2 Model Training and Experimentation (Google Colab)

For training and experimentation, Google Colab offers the required computational power, eliminating the need for a high-performance local machine. The free resources provided by Google Colab are sufficient to train Vision Transformers for this task. However, if we have access to Google Colab Pro, it provides increased computational limits, faster GPUs, and longer runtimes.

5.1.2.3 Hardware Specifications for Google Colab:

- **GPU**: NVIDIA Tesla T4 or P100 (available in free Colab, with the option of faster GPUs on Colab Pro)
- **RAM**: 12 GB (expandable to 25 GB on Colab Pro)
- **Disk Space**: Sufficient temporary storage for datasets and model checkpoints

Google Colab's cloud-based resources are essential for handling the computational demands of Vision Transformer models, which require significant memory and processing power for training.

5.1.2.4 Recommended Computer Specifications

In addition to using Google Colab for model training, we need a computer with adequate specifications to build, test, and deploy the Flutter-based mobile application. Here are the recommended computer specifications for developing the application smoothly, especially when using Android Studio or Xcode for testing and debugging.

1. **Processor**:
   - **Minimum**: Dual-core processor (Intel Core i3 or AMD equivalent)
   - **Recommended**: Quad-core processor or higher (Intel Core i5/i7 or AMD Ryzen 5/7)
2. **RAM**:
   - **Minimum**: 8 GB
   - **Recommended**: 16 GB or more
3. **Storage**:
   - **Minimum**: 256 GB SSD
   - **Recommended**: 512 GB SSD or more
4. **Operating System**:
   - **For Android Development**: Windows 10/11, macOS 10.15 or later, or Linux (Ubuntu 18.04 or later)
   - **For iOS Development**: macOS 10.15 or later (required to run Xcode and build for iOS)

5.1.3 Communication Standards

5.1.3.1 Application Development

For developing the mobile application, Flutter was chosen for its cross-platform capabilities. Flutter enables the development of a single codebase that runs on both Android and iOS devices, simplifying the development process.

- **Flutter**: The main framework for building the user interface and handling the application's interactive features.
- **Dart**: The programming language used with Flutter for developing the app's frontend logic.

- **TensorFlow Lite Interpreter**: Enables TensorFlow Lite model inference within the mobile application, allowing the app to perform rose color classification on-device without requiring an internet connection.
- **Text-to-Speech (TTS) Plugin**: For providing audio feedback, ensuring that visually impaired users can receive the color detection results through audio output.

5.2 Impact on Society, Environment and Sustainability

5.2.1 Impact on Life

The proposed mobile app for rose color detection aims to significantly enhance the quality of life for visually impaired individuals by providing them with an accessible and intuitive tool to experience and identify colors independently. Colors, especially in ornamental flowers like roses, play a vital role in emotional expression, cultural traditions, and social interactions. For individuals who are blind or visually impaired, the inability to perceive and appreciate colors can create a disconnect from these experiences.

By enabling real-time color detection through a smartphone, the app empowers users to actively engage with their environment and participate in activities where color recognition is important. This technology allows visually impaired individuals to experience the vibrancy and diversity of rose colors, enhancing their sensory interaction and fostering inclusivity in everyday life. Beyond aesthetics, the app has potential applications in education, where users can learn about color theory and its significance, further broadening their understanding and appreciation of the visual world.

Additionally, the app reduces the dependence on others for identifying colors, promoting autonomy and self-reliance. This independence has broader psychological benefits, including increased confidence and a sense of empowerment, as users gain the ability to explore their environment and make informed decisions based on real-time feedback from the app.

5.2.2 Impact on Society & Environment

The societal impact of a mobile-based rose color detection solution extends beyond individual benefits, fostering inclusivity and accessibility in technology. The app highlights the growing importance of designing assistive technologies that address the

needs of marginalized groups, particularly individuals with visual impairments. By creating a tool that enables visually impaired users to access color information, the app contributes to societal goals of equity, inclusion, and accessibility.

From an environmental perspective, the app indirectly supports sustainable practices by promoting digital solutions over physical color-matching tools or other resource-intensive alternatives. Unlike traditional methods, which may involve printing color charts or manufacturing additional hardware, the app leverages existing smartphone technology, reducing the need for additional physical resources. This aligns with sustainability goals by minimizing waste and utilizing digital infrastructure to deliver a powerful, scalable solution.

Furthermore, the app promotes the appreciation and conservation of roses, a symbolically significant and culturally valued plant. By enabling users to identify and appreciate the diversity of rose colors, the app may inspire greater interest in horticulture and plant conservation. This awareness could encourage sustainable gardening practices and support the preservation of biodiversity in ornamental plants.

In summary, the app's societal impact lies in its ability to empower visually impaired individuals, promote inclusivity, and raise awareness about the importance of accessible technology. Environmentally, it leverages existing digital infrastructure to minimize waste and foster sustainable practices, contributing to a more equitable and eco-friendly future.

5.2.3 Ethical Aspects

The development and deployment of a mobile app for rose color detection tailored for visually impaired individuals involve several ethical considerations. Foremost among these is ensuring inclusivity and equity in design. The app must be accessible to users across various demographics, including those with limited technological proficiency or access to high-end devices. This aligns with the ethical principle of providing equal opportunities for all individuals, regardless of socioeconomic status.

Another critical ethical aspect is data privacy and security. The app may process images from users' environments, raising concerns about the potential misuse of sensitive data. Ensuring that the app does not store, share, or misuse user-generated content is paramount. Secure data handling protocols and compliance with privacy regulations such as the General Data Protection Regulation (GDPR) must be prioritized to safeguard user trust.

Transparency in model design and output is also an essential ethical consideration. For visually impaired users, the app's feedback must be clear, interpretable, and actionable. The use of explainable AI techniques can help users understand how the app identifies colors, fostering trust and reducing reliance on "black box" systems. Finally, the app must avoid reinforcing biases, such as providing inaccurate results for certain skin tones or lighting conditions, ensuring it works effectively across diverse environments and populations.

5.2.4 Sustainability Plan

To ensure the app's long-term success and sustainability, a multi-faceted approach is essential, focusing on economic viability, environmental responsibility, and technological adaptability.

1. **Economic Sustainability:** A freemium model can be adopted, where the core functionalities are free, ensuring accessibility for all users, while advanced features are available through a subscription or one-time fee. Partnerships with non-profits and accessibility-focused organizations can provide funding to subsidize the app for low-income users. Regular updates and user feedback mechanisms will ensure the app remains relevant and effective, maintaining its user base and financial stability.

2. **Environmental Sustainability**: By leveraging existing smartphone technology, the app minimizes the need for additional hardware, reducing its environmental footprint. Continuous optimization of the app to reduce energy consumption and storage requirements will further support eco-friendly practices. Cloud-based updates can minimize waste associated with physical distribution, ensuring a scalable and sustainable solution.

3. **Technological Sustainability**: Regular updates to the app's machine learning model and architecture will ensure it remains compatible with advancements in smartphone and IoT technology. Incorporating new datasets and retraining the model to adapt to evolving environmental conditions and user needs will maintain its accuracy and usability over time. Open-source contributions or collaborations with research institutions can also ensure the app evolves with cutting-edge technological advancements.

5.3 Project Management and Financial Analysis

5.3.1 Risk Management

Risk management was a crucial part of this project to ensure that potential challenges were anticipated, mitigated, and monitored throughout the development process. Identifying and addressing risks proactively helped maintain the project timeline, budget, and quality of outcomes. The following categories of risks were identified, with strategies developed to minimize their impact on the project.

5.3.2 Technical Risks

Technical risks involve challenges related to model training, application development, and deployment. These risks could disrupt the project's development process and impact its overall functionality and performance.

- **Risk of Model Underperformance**: The Vision Transformer model might fail to achieve the required accuracy or robustness, particularly in differentiating similar rose colors under various lighting conditions. To mitigate this, extensive experimentation with different ViT configurations was conducted to select the optimal model. Additionally, data augmentation techniques were applied to increase dataset diversity and improve the model's generalization ability.

- **Limitations of Google Colab**: Although Google Colab provided the necessary resources for training, limitations such as restricted session times and memory usage posed potential challenges. To mitigate this, Google Colab Pro was considered for extended sessions and increased computational power. Model checkpoints were frequently saved by Google Drive to prevent data loss during unexpected interruptions.

5.3.3 Data Risks

Data-related risks could impact model training quality and generalization capabilities, as well as the app's reliability in real-world conditions.

- **Data Quality and Insufficiency**: The dataset quality plays a critical role in the performance of the Vision Transformer model. Insufficient data or poor-quality images may hinder the model's ability to distinguish rose colors accurately. To manage this risk, a high-resolution dataset was collected, and data augmentation was applied to expand the dataset and capture variations in lighting, angles, and orientations, ensuring the model received a representative sample of rose images.

- **Data Storage and Accessibility**: Reliance on Google Drive and Colab could introduce risks of data inaccessibility or storage limitations. To address this, a backup storage system was used for critical datasets, and Google Drive was monitored to ensure sufficient storage space. Data was organized and documented to ensure easy access during development.

5.3.4 Financial Risks

Financial constraints can limit the resources available for experimentation, testing, and deployment. Managing the budget efficiently was essential to ensure the project's successful completion without compromising quality.

- **Cost of Additional Computational Resources**: Upgrading to Google Colab Pro, purchasing additional device testing resources, or investing in cloud storage could introduce unexpected costs. To mitigate this risk, a budget was allocated specifically for Google Colab Pro and potential cloud storage needs, with cost-monitoring tools used to track expenditures. Free or low-cost software and hardware resources were prioritized where possible.

- **Potential Increase in Development Time**: Financial constraints could result in delays if further testing or troubleshooting is needed. The project timeline was planned with buffer periods to account for unforeseen challenges. Tasks were prioritized based on their impact on the final product, and any additional costs were weighed carefully against their necessity for achieving project goals.

5.3.5 Financial Analysis

The financial analysis for this project focused on optimizing costs while ensuring high performance and accessibility. By utilizing cloud-based resources, open-source tools, and existing devices, the project was able to manage expenses effectively. The budget was divided across several key areas, including cloud computing for model training, software, hardware for testing, and application development.

Table 5.1: Financial Analysis

Expense Category	Description	Cost Impact
Cloud Computing Costs	Training Vision Transformer models on Google Colab; Google Colab Pro subscription for enhanced computational efficiency and extended session times.	

Software Costs	Open-source and academic-licensed software used (Python, TensorFlow, Flutter, OpenCV) to minimize costs.	
Hardware and Testing Devices	Google Colab provided training resources; testing conducted on personal or university-provided Android and iOS devices, avoiding additional hardware expenses.	
Application Development	Developed using Flutter for cross-platform support on Android and iOS, reducing development time and platform-specific costs.	
Miscellaneous Expenses	Minor expenses for internet connectivity, Google Drive storage for dataset management, and potential future cloud storage needs.	

5.4 Complex Engineering Problem

5.4.1 Complex Problem Solving

Table 5.1 provides a detailed mapping of the research problem to the problem-solving categories. It demonstrates how the project addresses key aspects such as depth of knowledge, conflicting requirements, and stakeholder involvement.

Table 5.2: Mapping with complex problem-solving.

EP1 Dept of Knowledge	EP2 Range Of Conflicting Requirements	EP3 Depth of Analysis	EP4 Familiarity of Issues	EP5 Extent of Applicable Codes	EP6 Extent Of Stake-holder Involvement	EP7 Interdependence
Deep understanding of different Vision transformer models(Vit-16, Vit-32) for disease detection	Balancing accuracy, computational efficiency, and dataset quality	Evaluating models using accuracy, F1-score, and recall metrics	Addressing dataset limitations and scalability issues	Following best practices in TensorFlow and PyTorch usage	Farmers and agricultural experts as primary beneficiaries	Integration of preprocessing, training, and evaluation workflows

Mapping with Knowledge Profile for EP1

Table 5.2 maps the Depth of Knowledge (EP1) to the Knowledge Profile categories. It illustrates the application of engineering fundamentals, advanced techniques, and research literature in the project.

Table 5.3: Mapping with knowledge Profile.

K3 Engineering Fundamentals	K4 Specialist Knowledge	K5 Engineering Design	K6 Engineering Practice	K8 Research Literature
Application of machine learning and computer vision principles	Advanced techniques like ViT models	Workflow design from data preprocessing to evaluation	Implementation using cloud-based Google Colab platform	Building the foundation through an extensive literature review

5.4.2 Engineering Activities

This section provides a mapping with engineering activities. Each mapping highlights the activities undertaken as part of the research and provides a rationale for their inclusion.

Table 5.3 highlights the complex engineering activities involved in the research, such as utilizing cloud resources, fostering collaboration, introducing innovative hybrid models, and addressing societal and environmental impacts. It emphasizes the familiarity with cutting-edge frameworks.

Table 5.4: Mapping with complex engineering activities.

EA1 Range of Resources	EA2 Level of Interaction	EA3 Innovation	EA4 Consequences for Society and Environment	EA5 Familiarity
Utilization of Google Colab's cloud-based GPU resources for efficient model training.	Collaboration with agricultural experts for real-world validation.	Integration of ViT models for innovative solutions.	Reduction in pesticide overuse and environmental harm.	Familiarity with TensorFlow and PyTorch frameworks.

5.5 Summary

This chapter explored the broader implications of a mobile app for rose color detection for

visually impaired individuals, highlighting its societal, environmental, and ethical impacts. The app empowers users by providing autonomy and inclusivity, while promoting sustainability through its reliance on existing technologies and eco-friendly practices. Ethical considerations such as data privacy, transparency, and equitable design are critical to ensuring the app's acceptance and effectiveness. A comprehensive sustainability plan outlines strategies for maintaining the app's economic, environmental, and technological viability. Together, these aspects underscore the app's potential to create meaningful, long-lasting impacts on individuals' lives, society, and the environment.

Chapter 6

Conclusion

6.1 Summary

1 This study presents the development of a mobile app-based solution for rose color detection tailored to visually impaired individuals. By leveraging advanced machine learning techniques, including Vision Transformers (ViTs) and hybrid CNN-ViT architectures, the proposed system demonstrates significant potential to address the challenges of real-time color recognition in diverse environments. The app aims to enhance user independence by providing intuitive, accurate, and actionable feedback on rose colors, fostering inclusivity and accessibility.

2 The integration of lightweight architectures ensures compatibility with standard mobile devices, making the app feasible for widespread use. Techniques like synthetic data generation using GANs were employed to address the limitations of labeled datasets, improving model generalization and performance. The app's adaptability to various lighting and background conditions further ensures its reliability in real-world scenarios. [Mention specific accuracy here].

3 Overall, this research contributes to the growing field of assistive technologies by providing a practical, scalable, and user-centric solution. It bridges the gap between technological advancements in machine learning and the needs of individuals with visual impairments, promoting inclusivity, accessibility, and empowerment.

6.2 Limitation

While the study demonstrates significant progress, certain limitations should be acknowledged:

1. **Dataset Limitations**: The model's performance is constrained by the diversity and volume of the dataset. Although synthetic data generation was used to mitigate this, further real-world data collection is necessary to enhance the model's adaptability. [Mention specific accuracy limitations].

2. **Hardware Constraints**: Despite the app's lightweight architecture, performance on extremely low-end devices may still be affected. Optimization efforts must continue to address these challenges, particularly for users in resource-constrained settings.

3. **User Feedback and Testing**: The app's usability and effectiveness have not been extensively tested with visually impaired users. Comprehensive user testing and iterative design improvements are essential for ensuring the app meets its intended objectives.

4. **Generalizability Across Contexts**: While the app is designed specifically for rose color detection, its adaptability to other objects or environments has not been tested. Additional development and validation would be required for broader applications.

5. **Conflict of Interests**: There are no conflicts of interest reported in this study. All efforts have been made to conduct the research impartially and with a focus on promoting inclusivity and accessibility for visually impaired individuals.

6.3 Future Work

Future research and development efforts can build upon the current study in the following ways:

1. **Enhancing Dataset Diversity**: Expanding the dataset to include a broader range of rose colors, lighting conditions, and environmental contexts can further improve the model's robustness and adaptability. Collaborative data collection efforts with botanical gardens and rose cultivators could enrich the dataset.

2. **Improving Model Accuracy and Efficiency**: Continued refinement of the model's architecture to balance accuracy and computational efficiency is crucial. Exploring advanced optimization techniques, such as pruning, quantization, or distillation,

could enable better performance on low-power devices without compromising accuracy. [Mention potential accuracy targets or improvements].

3. **Personalized Feedback Mechanisms**: Incorporating features that allow users to customize the app's feedback, such as auditory descriptions or tactile outputs, could enhance the app's usability for diverse user preferences and needs.

4. **Integration with Assistive Ecosystems**: Extending the app's functionality to integrate with other assistive technologies, such as voice assistants or wearable devices, could create a more comprehensive support system for visually impaired individuals.

5. **Cross-Application Development**: Adapting the app for additional use cases, such as identifying colors in broader contexts (e.g., clothing or paintings), could increase its utility and market appeal while benefiting a larger user base.

References

[References should be in IEEE format]

[1] Saini, A., Guleria, K., & Sharma, S. (2024, July). A ViT Vision Transformer Model for Rose Leaf Disease Classification. In 2024 2nd World Conference on Communication & Computing (WCONF) (pp. 1-4). IEEE.

[2] Keerthan Bhat, H., Mukund, A., Nagaraj, S., & Prakash, R. (2022, November). LeafViT: Vision Transformers-Based Leaf Disease Detection. In International Conference on Innovations in Computational Intelligence and Computer Vision (pp. 85-102). Singapore: Springer Nature Singapore.

[3] Thakur, P. S., Chaturvedi, S., Khanna, P., Sheorey, T., & Ojha, A. (2023). Vision transformer meets convolutional neural network for plant disease classification. Ecological Informatics, 77, 102245.

[4] Khaleel, M. I., Sai, P. G., Kumar, A. U. R., Raja, P., & Hoang, V. T. (2022, April). Rose plant leaves: Disease detection and pesticide management using cnn. In 2022 6th International conference on trends in electronics and informatics (ICOEI) (pp. 1067-1072). IEEE.

[5] Mustofa, S., Munna, M. M. H., Emon, Y. R., Rabbany, G., & Ahad, M. T. (2023). A comprehensive review on Plant Leaf Disease detection using Deep learning. arXiv preprint arXiv:2308.14087.

[6] Ahmed, F., Ahad, M. T., & Emon, Y. R. (2023). Machine Learning-Based Tea Leaf Disease Detection: A Comprehensive Review. arXiv preprint arXiv:2311.03240.

[7] Bhuyan, P., & Singh, P. K. (2024, January). Evaluating Deep CNNs and Vision Transformers for Plant Leaf Disease Classification. In International Conference on Distributed Computing and Intelligent Technology (pp. 293-306). Cham: Springer Nature Switzerland.

[8] Bhowmik, A. C., Ahad, M. T., Emon, Y. R., Ahmed, F., Song, B., & Li, Y. (2024). A customised Vision Transformer for accurate detection and classification of Java Plum leaf disease. Smart Agricultural Technology, 8, 100500.

[9] Gangwar, A., Dhaka, V. S., Rani, G., Khandelwal, S., Zumpano, E., & Vocaturo, E. (2024). Time and Space Efficient Multi-Model Convolution Vision Transformer for Tomato Disease Detection from Leaf Images with Varied Backgrounds. Computers, Materials & Continua, 79(1).

[10] Hossain, M. A., Sakib, S., Abdullah, H. M., & Arman, S. E. (2024). Deep learning for

mango leaf disease identification: A vision transformer perspective. Heliyon, 10(17).
[11] Singh, A. K., Rao, A., Chattopadhyay, P., Maurya, R., & Singh, L. (2024). Effective plant disease diagnosis using Vision Transformer trained with leafy-generative adversarial network-generated images. Expert Systems with Applications, 124387.
[12] Barman, U., Sarma, P., Rahman, M., Deka, V., Lahkar, S., Sharma, V., & Saikia, M. J. (2024). Vit-SmartAgri: vision transformer and smartphone-based plant disease detection for smart agriculture. Agronomy, 14(2), 327.
[13] Sundaraj, A., Isravel, D. P., & Dhas, J. P. M. (2024, April). Diagnosis of Plant Leaf Disease using Vision Transformer. In 2024 10th International Conference on Communication and Signal Processing (ICCSP) (pp. 82-87). IEEE.
[14] You, H., Lee, K., Oh, J., & Lee, E. C. (2023). Efficient and low color information dependency skin segmentation model. Mathematics, 11(9), 2057.
[15] Sawarkar, V., & Kawathekar, S. (2018). A review: Rose plant disease detection using image processing. IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN, 2278-0661.
[16] Kusuma, S., & Jothi, K. R. (2024). Early betel leaf disease detection using vision transformer and deep learning algorithms. International Journal of Information Technology, 16(1), 169-180.
[17] Thai, H. T., Le, K. H., & Nguyen, N. L. T. (2023). FormerLeaf: An efficient vision transformer for Cassava Leaf Disease detection. Computers and Electronics in Agriculture, 204, 107518.
[18] Thai, H. T., Tran-Van, N. Y., & Le, K. H. (2021, October). Artificial cognition for early leaf disease detection using vision transformers. In 2021 International Conference on Advanced Technologies for Communications (ATC) (pp. 33-38). IEEE.
[19] Chen, Z., Wang, G., Lv, T., & Zhang, X. (2024). Using a Hybrid Convolutional Neural Network with a Transformer Model for Tomato Leaf Disease Detection. Agronomy, 14(4), 673.
[20] Ahmed, F., Emon, Y. R., Ahad, M. T., Munna, M. H., & Mamun, S. B. (2023, November). A Fuzzy-Based Vision Transformer Model for Tea Leaf Disease Detection. In International Conference on Trends in Computational and Cognitive Engineering (pp. 229-242). Singapore: Springer Nature Singapore.
[21] Salamai, A. A., Ajabnoor, N., Khalid, W. E., Ali, M. M., & Murayr, A. A. (2023). Lesion-aware visual transformer network for Paddy diseases detection in precision agriculture. European Journal of Agronomy, 148, 126884.

Rose color detection for blind people's using deep learning.

ORIGINALITY REPORT

14%	9%	7%	9%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	1%
2	v1.overleaf.com Internet Source	1%
3	Mehdi Ghayoumi. "Generative Adversarial Networks in Practice", CRC Press, 2023 Publication	1%
4	Submitted to University of Greenwich Student Paper	<1%
5	arxiv.org Internet Source	<1%
6	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	<1%
7	Submitted to The University of the West of Scotland Student Paper	<1%
8	Kutub Thakur, Helen G. Barker, Al-Sakib Khan Pathan. "Artificial Intelligence and Large	<1%