A PROJECT REPORT

ON

RICE LEAF DISEASES CLASSIFICATION USING CNN WITH TRANSFER LEARNING

Submitted in partial fulfillment of the requirements for the award of the degree.

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Under the guidance of

Mr. PANDERETI PRAVEEN, M.Tech,

Assistant Professor / Computer Science and Engineering

BY

BALAPURAM JAYANTH	**21751A0523**
A C GIRISH	**22755A0501**
D KANNAN	**21751A0544**
D CHARAN KUMAR	**21751A0550**

SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES, CHITTOOR-517127, A.P.

(Autonomous)

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

(2021-2025)

SREENIVASA INSTITUTE OF TECHNOLOGYAND MANAGEMENT STUDIES, Chittoor, A.P.-517127.

(Autonomous)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIED CERTIFICATE

This is to certify that the project work entitled **"RICE LEAF DISEASES CLASSIFICATION USING CNN WITH TRANSFER LEARNING"** is a genuine work of

BALAPURAM JAYANTH	**21751A0523**
A C GIRISH	**22755A0501**
D KANNAN	**21751A0544**
D CHARAN KUMAR	**21751A0550**

Submitted to the department of Computer Science and Engineering, in a partial fulfilment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGYN in COMPUTER SCIENCE AND ENGINEERING from Jawaharlal Nehru Technological University Ananthapur, Ananthapuramu.

Signature of the Supervisor
Mr. PANDERETI PRAVEEN, M.Tech.,
Assistant Professor,
Department of Computer Science And Engineering,
Sreenivasa Institute of Technology and Management Studies, Chittoor, A.P.

Signature of the Head of Department
Dr. KARUNIA KRISHNA PRIYA.,
Associate Professor & HOD
Department of Computer Science And Engineering,
Sreenivasa Institute of Technology and Management Studies, Chittoor, A.P.

Submitted for University Examination on (Viva-Voce) held on ………………………

INTERNAL EXAMINER **EXTERNAL EXAMINER**

SREENIVASA INSTITUTE OF TECHNOLOGYAND MANAGEMENT STUDIES, Chittoor, A.P.-517127.

(Autonomous)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Institute Vision

To emerge as a Centre of Excellence for Learning and Research in the domains of engineering, computing and management.

Institute Mission

M1: Provide congenial academic ambience with state-art of resources for learning and research.

M2: Ignite the students to acquire self-reliance in the latest technologies.

M3: Unleash and encourage the innate potential and creativity of students.

M4: Inculcate confidence to face and experience new challenges.

M5: Foster enterprising spirit among students.

Department Vision

To contribute for the society through excellence in Computer Science and Engineering with a deep passion for wisdom, culture and values.

Department Mission

- Provide congenial academic ambience with necessary infrastructure and learning resources.
- Inculcate confidence to face and experience new challenges from industry and society.
- Ignite the students to acquire self-reliance in the latest technologies.
- Foster Enterprising spirit among students.

SREENIVASA INSTITUTE OF TECHNOLOGYAND MANAGEMENT STUDIES, Chittoor, A.P.-517127.

(Autonomous)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PROGRAMME EDUCATIONAL OBJECTIVES (PEO's):

PEO1: Excel in Computer Science and Engineering program through quality studies, enabling success in computing industry. **(Professional Competency)**.

PEO2: Surpass in one's career by critical thinking towards successful services and growth of the organization, or as an entrepreneur or in higher studies. **(Successful Career Goals)**.
PEO3: Enhance knowledge by updating advanced technological concepts for facing the rapidly changing world and contribute to society through innovation and creativity **(Continuing Education and Contribution to Society)**.

PROGRAM SPECIFIC OUTCOMES (PSO's):

PSO1: Have Ability to understand, analyses and develop computer programs in the areas like algorithms, system software, web design,big data analytics, and networking.
PSO2: Deploy the modern computer languages, environment, and platforms in creating innovative products and solutions.

PROGRAMME OUTCOMES (PO's):

PO1 - Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2 - Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3 - Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4 - Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions. **PO5 - Modern tool usage:** Create,

select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6 - The engineer and society: Apply reasoning informed by the

contextual

knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7 - Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8 - Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9 - Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10 - Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11 - Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12 - Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

SREENIVASA INSTITUTE OF TECHNOLOGYAND MANAGEMENT STUDIES, Chittoor, A.P.-517127.

(Autonomous)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Course Out Comes for Project Work

On completion of project work the student will be able to

CO1. Demonstrate in-depth knowledge on the project topic.

CO2. Identify, analyze and formulate complex problem chosen for project work to attain substantiated conclusions.

CO3. Design solutions to the chosen project problem.

CO4. Undertake investigation of project problem to provide valid conclusions.

CO5. Use the appropriate techniques, resources and modern engineering tools necessary for project work.

CO6. Apply project results for sustainable development of the society.

CO7. Understand the impact of project results in the context of environmental sustainability.

CO8. Understand professional and ethical responsibilities while executing the project work.

CO9. Function effectively as individual and a member in the project team.

CO10. Develop communication skills, both oral and written for preparing and presenting project report.

CO11. Demonstrate knowledge and understanding of cost and time analysis required for carrying out the project.

CO12. Engage in lifelong learning to improve knowledge and competence in the chosen area of the project.

SREENIVASA INSTITUTE OF TECHNOLOGYAND MANAGEMENT STUDIES, Chittoor, A.P.-517127.

(Autonomous)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

<u>CO – PO MAPPING</u>

CO\PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO.1	3	-	-	-	-	-	-	-	-	-	-	-	3	3
CO.2	-	3	-	-	-	-	-	-	-	-	-	-	3	3
CO.3	-	-	3	-	-	-	-	-	-	-	-	-	3	3
CO.4	-	-	-	3	-	-	-	-	-	-	-	-	3	3
CO.5	-	-	-	-	3	-	-	-	-	-	-	-	3	3
CO.6	-	-	-	-	-	3	-	-	-	-	-	-	3	3
CO.7	-	-	-	-	-	-	3	-	-	-	-	-	3	3
CO.8	-	-	-	-	-	-	-	3	-	-	-	-	3	3
CO.9	-	-	-	-	-	-	-	-	3	-	-	-	3	3
CO.10	-	-	-	-	-	-	-	-	-	3	-	-	3	3
CO.11	-	-	-	-	-	-	-	-	-	-	3	-	3	3
CO.12	-	-	-	-	-	-	-	-	-	-	-	3	3	3
CO	3	3	3	3	3	3	3	3	3	3	3	3	3	3

SREENIVASA INSTITUTE OF TECHNOLOGYAND MANAGEMENT STUDIES, Chittoor, A.P.-517127.

(Autonomous)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Ananthapuramu)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Evaluation Rubrics for Project Work

Rubric (CO)	Excellent (wt = 3)	Good (wt = 2)	Fair (wt = 1)
Selection of Topic (CO1)	Selected a latest topic through complete knowledgeof facts and Concepts	Selected a topic through partial knowledge off acts and concepts	Selected at opicthrough improper knowledge of facts and concepts
Analysis and Synthesis (CO2)	Thorough comprehensionthrough analysis/ synthesis	Reasonable comprehensionthrough analysis/ synthesis	Improper comprehensionthrough analysis/ synthesis
Problem Solving (CO3)	Thorough comprehension about what is proposed in the literature papers	Reasonable comprehension about what is proposed in the literature papers	Improper comprehension about what is proposed in the literature
Literature Survey (CO4)	Extensive literature surveywith standard References	Considerable literature survey with standard References	Incomplete literature survey with substandard References
Usage of Techniques &Tools (CO5)	Clearly identified and has complete knowledge of techniques & tools used in the project work	Identified and has sufficient knowledge of techniques & tools used in the project work	Identified and has inadequate knowledge of techniques & tools used in project work
Project work impact on Society (CO6)	Conclusion of project work has strong impact on society	Conclusion of project work has considerable impact on society	Conclusion of project work has feeble impact on society
Project work impact on Environment (CO7)	Conclusion of project work hasstrong impact on Environment	Conclusion of project work has considerable impact on environment	Conclusion of project work has feeble impact on environment
Ethical attitude (CO8)	Clearlyunderstands ethical and social practices.	Moderateunderstanding of ethical and social practices.	Insufficient understandingof ethical and social practices.
Independent Learning (CO9)	Did literature survey and selected topic with little Guidance	Did literature survey and selected topic with considerable guidance	Selected a topic as suggested by the Supervisor
Oral Presentation (CO10)	Presentation in logical sequence with key points, clear conclusion and excellent language	Presentation with key points, conclusion and good language	Presentation with insufficient key points and improper Conclusion
Report Writing (CO10)	Status report with clear and logical sequence of chapters using excellent language	Status report with logical sequence of chapters using understandable language	Status report not properlyorganized
Time and Cost Analysis (CO11)	Comprehensivetime and cost analysis	Moderatetime and cost analysis	Reasonable time and cost analysis
Continuous learning (CO12)	Highly enthusiastic towards continuous Learning	Interested in continuous learning	Inadequate interest in continuous learning

ACKNOWLEDGEMENT

A Project of this magnitude would have not been possible without the guidance and co- ordination of many people. I am fortune in having top quality people to help, support and guide us in every step towards our goal.

Our team is very much grateful to the Chairman **Sri K. RANGANADHAM** Garu for his encouragement and stalwart support. We are also extremely indebted to the Secretary **Sri D.K. BADRI NARAYANA** Garu for his constant support.

We express our sincere thanks to our Academic Advisor **Dr. K.L. NARAYANA**, **M.Tech., Ph.D,** further, we would like to express our profound gratitude to our principal **Dr.N.VENKATACHALAPATHI, M.T.ech, Ph.D** for providing all possible facilities throughout the completion of our project work.

We express our sincere thanks to our Dean (Academics), **Dr.M.Saravanan, M.E., Ph.D.,** further we express our sincere thanks to our Head of the Department

Dr. KARUNIA KRISHNA PRIYA., for her co-operation and valuable suggestions towards the completion of project work.

We express our sincere thanks to our guide **Mr. PANDERETI PRAVEEN, M.Tech.,**for offering us the opportunity to do this work under his guidance.

We express our sincere salutation to all other teaching and non-teaching staff of our department for their direct and indirect support given during our project work. Last but not the least, we dedicate this work to our parents and the Almighty who have been with us throughout and helped us to overcome the hard times.

DECLARATION

We certify that

- The work contained in this report is original and has been done by me under the Guidance of my supervisor.
- The work has not been submitted to any other Institute for any degree or diploma.
- We have followed the guidelines provided by the Institute in preparing the report.
- We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
- Whenever we have used materials (data, theoretical analysis, figures, and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

BALAPURAM JAYANTH	**21751A0523**
A C GIRISH	**22755A0501**
D KANNAN	**21751A0544**
D CHARAN KUMAR	**21751A0550**

ABSTRACT

Rice is one of the key crops cultivated in India that suffers from numerous diseases at different stages of its cultivation. It is extremely challenging for the farmers to identify these diseases manually with their limited knowledge. Recent advances in Deep Learning indicate that Automatic Image Recognition systems based on Convolutional Neural Network (CNN) models can be extremely helpful in such issues. Because rice leaf disease image dataset is hard to get, we have constructed our own small-sized dataset for which we used Transfer Learning in order to train our deep learning model. The suggested CNN structure is built on VGG16 and has been trained and tested on the data gathered from the rice fields as well as from the internet.

KEYWORDS: Convolutional Neural Network, VGG16, Resnet-50, CNN, Deep Learning, Fine-Tuning, Rice Leaf Diseases, Transfer Learning.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1. OBJECTIVE

Our contribution in applying Convolutional Neural Networks (CNNs) to classify rice leaf disease using Transfer Learning is pioneering. Through the utilization of deep learning and the models such as VGG16, you have bridged the gap of dataset size limitation, opening doors to precise disease detection in rice plants. Your study not only benefits farmers but also demonstrates the capability of AI to transform agriculture with improved crop management. Your commitment to innovation and problem-solving in agriculture with technology is inspiring and has significant potential for impactful change.

1.2. PROBLEM STATEMENT

Manual diagnosis of diseases infecting rice plants in India is a serious issue for farmers since there is scarce expertise. For this, drawing on Deep Learning and Transfer Learning, the current research is directed towards building a robust automatic image classification system on a tailored dataset. The project is designed around rice leaf disease, working on a transformed VGG16 structure, which was trained and tested on a set of curated data that included field images and internet images. This study aims at accurate disease classification for supporting farmers in early intervention and detection.

1.3. OBJECTIVE OF THE PROJECT

The aim of this research work is to utilize Convolutional Neural Networks (CNNs) with Transfer Learning, using VGG16 architecture, for precise classification of rice leaf diseases. Even with sparse access to a small-scale proprietary dataset, this study seeks to utilize Transfer Learning methodology to successfully train and test the model on a collection of images obtained from rice fields and the internet. The objective is to create an automated system that is capable of identifying and classifying different diseases inflicting rice plants with proficiency, supporting farmers in accurate and timely disease detection.

1.4. SCOPE

The scope of the study proposal is utilizing Transfer Learning with a VGG16-based Convolutional Neural Network to diagnose rice leaf diseases. Utilizing a dataset generated from a mixture of field and internet sources, the model will be able to correctly diagnose multiple rice plant diseases. The emphasis is on breaking restricted data limitations using Transfer Learning approaches to fine-tune the pre-trained model in efficient disease recognition, which might assist farmers with early diagnosis and control of the problems.

CHAPTER 2

LITERATURE SURVEY

[1] R. R. Atole, D. Park, "A Multiclass Deep Convolutional Neural Network Classifier for Detection of Common Rice Plant Anomalies," International Journal Of Advanced Computer Science And Applications,vol. 9,no. 1,pp. 67–70,2018.

This research discusses the application of deep convolutional neural network in classifying rice plants based on health status from images of its leaves. A three-class classifier was used representing healthy, unhealthy, and snail-infested plants through transfer learning from a deep network AlexNet. The network had an accuracy of 91.23%, utilizing stochastic gradient descent with mini batch size of thirty (30) and initial learning rate of 0.0001. Six hundred (600) rice plant images representing the classes were utilized for training. The training and testing dataset-images were taken from rice fields surrounding the district and approved by field-level agriculture technicians.

[2] P. Konstantinos Ferentinos, "Deep Learning Models for Plant Disease Detection and Diagnosis," Computers and Electronics in Agriculture,vol. 145,pp. 311–318,2018.

Convolutional neural network models were trained in this paper to conduct plant disease detection and diagnosis from simple leaves images of healthy and infected plants, utilizing deep learning methods.Training was conducted using an open database of 87,848 images with 25 plants in a set of 58 different classes of [plant, disease] combinations, including healthy plants. A number of model architectures were learned, with the optimal being a 99.53% success rate in recognizing the corresponding [plant, disease] pair (or healthy plant). The extremely high rate of success renders the model an extremely effective advisory or early warning facility, and a method that may be extended further for use in an integrated plant disease recognition system to function under actual cultivation conditions.

[3] Y. Lu, S. Yi,N. Zeng,Y. Liu, and Y. Zhang, "Identification of Rice Diseases Using Deep Convolutional Neural Networks," Neurocomputing, 267, pp. 378-384,2017.

A proper and timely identification of rice plant diseases and pests can make economic losses negligible. It can assist farmers in the application of timely treatment. Deep learning based convolutional neural networks (CNN) in recent times enabled the researchers to significantly enhance image classification accuracy. We introduce, in this paper, a deep learning based system for detecting pests and diseases of rice plants in real life scenario captured images having heterogeneous background. We have tested different state-of-the-art convolutional neural networks on our extensive dataset of rice diseases and pests with both inter-class and intra-class variations. The outcome demonstrates that we are able to effectively recognize and detect nine classes of rice diseases and pests including healthy plant class with a deep convolutional neural network, with maximum accuracy of 99.53% on test set.

[4] V. Singh, A. Misra, "Detection of Plant Leaf Diseases Using Image Segmentation and Soft Computing Techniques," Information Processing in Agriculture,vol. 4,no. 1,pp. 41–49,2017

Agricultural productivity is something on which economy highly depends. This is the one of the reasons that disease detection in plants plays an important role in agriculture field, as having disease in plants are quite natural. If there is not proper care in this section then it gives severe impacts on plants and because of which respective product quality, quantity or productivity is impacted. For example a disease little leaf disease is a risky disease occurring in pine trees in United States. Detection of plant disease by some automatic method is helpful as it eliminates a huge work of surveillance in large farms of crops, and at very initial stage itself identifies the disease symptoms i.e. when they occur on plant leaves. This paper introduces an algorithm for image segmentation technique which is applied for automatic detection and classification of plant leaf diseases. It also includes survey on various diseases classification methods that may be employed in plant leaf disease detection. Segmentation of the image, being a critical parameter for disease identification in plant leaf disease, is performed by utilizing genetic algorithm.

[5] Y. Es-saady,T. El Massi,M. El Yassa,D. Mammass, and A. Benazoun, "Automatic recognition of plant leaves diseases based on serial combination of two SVM classifiers," International Conference on Electrical and Information Technologies (ICEIT) pp. 561-566, 2016.

The paper proposes a machine vision system for automatic plant leaves disease recognition from images. The system relies on serial combination technique of two SVM classifiers. The first classifier bases its classification of the images using the color, and it factors in, in this stage, that the nearest or similar disease with the same class. Second, the second classifier is then applied to differentiate the classes having similar color using the shape and texture features. The experiments of this research are conducted on six disease classes of diseases such as three classes of pest insects damages (Leaf miners, Thrips and Tuta absoluta) and three classes of pathogens symptoms (Early blight, Late blight and Powdery mildew). The findings of the research demonstrate the benefits of the suggested approach over the other available approaches.

CHAPTER 3

PROJECT DESCRIPTION

Rice is the major food source in India as well as globally. It has been infested by many diseases at different stages of its growth. Thus, early diagnosis and treatment of such diseases are advantageous to have high quality and maximum quantity but this is extremely challenging because of the vast area of land under single farmers and the immense range of diseases and also the possibility of more than one disease in a single plant. Farming expertise is not available in remote locations and it is a time consuming process. Thus, the Automated Systems are needed. In order to alleviate the farmers' struggle and ensure better accuracy of disease detection in plants, work under research using different machine learning algorithms like Support Vector Machine(SVM), Artificial Neural Networks has been carried out.

But the correctness of such systems largely relies on feature selection methods. Recent works on convolutional neural networks have brought tremendous improvement in image based recognition by making image preprocessing unnecessary as well as offering inbuilt feature selection. Another issue is that it is extremely hard to get large sized dataset for such issues. For cases where size of the dataset is relatively small, it is more preferable to use a model which is pretrained on a large dataset. This is referred to as Transfer Learning and can be used to develop a model that can be employed as a fixed feature extractor by stripping the last fully connected layer or by fine-tuning the last several layers that will be more specific to the involved dataset.

Now mobile phones are in everyone's hands and thus we have thought of an automated process where the farmer can upload the image of the diseased leaf and share it with our server where the neural network will identify the disease and the disease type along with the cure can be returned to the farmer. In this paper we have suggested the architecture of the disease classification component of the automated system. Drawing inspiration from the research on convolutional neural networks, in this paper, we have established the deep learning solution on our rice disease dataset that we have gathered over the last few months. We have utilized the pre-trained VGG16 model (Trained on the massive ImageNet data) and utilizing Transfer Learning we have finetuned the fully connected layers so that we can use our own dataset and at the end we have performed some error analysis and tried to describe the cause of the error.

Plant Disease Detection using CNN

CNN is trained with 87,848 images, of which 25 plant varieties have 58 classes including healthy plants. Different models were trained, of which the best one gave 99.53% accuracy in proper identification. CNN was utilised to train 54306 images of 14 crop types and 26 diseases and healthy leaves. The success rate of 99.35% declined to 31.4% while tested on another dataset gathered from real life situation. In disease severity problem and why it is more difficult than disease classification is explained. High intra-class similarity between the different images of same class makes it even harder to identify.

Rice Disease Detection using CNN

Convolutional neural network classifier is applied on a collection of 227 images of snail-bitten, diseased and healthy rice plants. The classifier is transfer learning based on AlexNet. The above architecture is trained and 91.23% accuracy is achieved but the architecture can only predict if the plant is diseased or not. The authors gathered 500 images of 10 various diseases of leaf and stem of rice. They created an architecture based on Le-Net and AlexNet and obtained 95.48% on test set. Because data is highly sparse they have employed different preprocessing step such as resizing image to 512*512, normalization, PCA and whitening. They employed stochastic pooling in place of max pooling and mentioned that it avoids over fitting.

Rice disease types and dataset description

The rice image dataset has been gathered during recent months primarily from the fields of cultivation of Madarat village (District: South 24 Parganas) of Baruipur, Dharinda village (District: Purba Medinipur) of Tamluk and Basirhat (District North 24 Parganas), a part of West Bengal state in India and the Internet. Photographs were shot by Motorola E4 Plus and Redmi 5A phone camera. The symptoms and information regarding the diseases have been gathered from the International Rice Research Institute (IRRI) Rice Knowledge Bank website. There were few images for training our system, so we have applied some data augmentation methods using Keras Documentation to obtain a significant number of images.

The data consists of 1649 images of rice diseased leaves comprising three most prevalent diseases i.e. Rice Leaf Blast, Rice Leaf Blight, and Brown Spot. There are 507 images of Healthy leaves. We have not performed any step for removing noise from raw data. There were a few issues encountered in gathering the data such as the lack of proper lighting and the presence of more than one disease on the same plant. We used image preprocessing techniques such as resizing and zooming to overcome them. The number of images that were possible to gather from the fields are very few for training CNN so we have employed a set of augmentation techniques such as zoom, horizontal and vertical shift, and rotation that are explained in the Implementation Section below. The following sections explain the classes of Rice Leaf diseases on which we have worked.

Fig. 1. (a)-(c) From Left to right. (a) Leaf Blast (b) Leaf Blight and (c) Brown Spot

A. Leaf Blast

It is a fungal disease due to Magnaporthe oryzae. The early symptoms are white to grey-green spots which are spindle-shaped or elliptical with dark red to brownish margins. Some are diamond shaped with wide centers and tapering ends. In the Figure 1 (a) spindle shaped lesions with white spots and dark brown border can be observed.

B. Leaf Blight

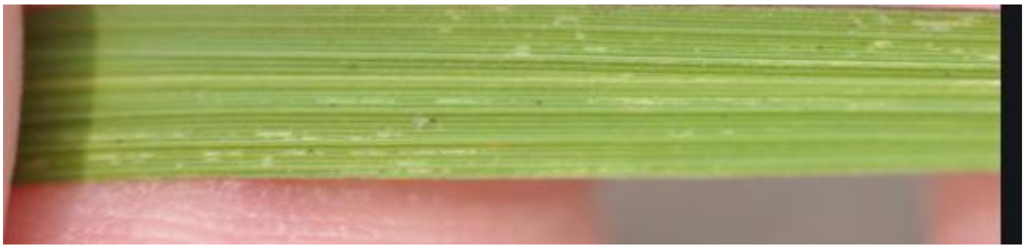
It is a bacterial infection caused by Xanthomonas oryzae. Leaves infected with this turn greyish green and roll up and subsequently yellow and straw colored before ultimately dying after wilting. Lesions have wavy edges and move towards the base. In young lesions, bacterial ooze like morning dew drop can be seen. Figure 1 (b) is of Leaf Blight affected leaves.

C.Brown Spot

It is a fungal disease. The infected leaves contain many large spots on the leaves that can kill the entire leaf. Small, circular, dark brown to purple-brown lesions are seen in the leaves at the early stage. Circular to oval, fully developed lesions are light brown to gray center with a reddish brown margin due to the toxin secreted by the fungi. Illustrated in Fig 1 (c) are the tiny dark brown lesions of the Brown Spot infected leaves.

D.Healthy

Image is healthy but it is categorized as Brown Spot likely because the image is blurred and contrast is poor.



CHAPTER

SYSTEM ANALYSIS

2.1. EXISTING SYSTEM

The current system used in Rice Leaf Disease Classification uses Convolutional Neural Network (CNN) algorithm to scan and classify different rice leaf diseases. Taking advantage of deep learning concepts, the system interprets the image data from rice leaves by drawing out salient features that correctly identify and classify diseases. Through this auto procedure, there is early disease detection and ability to intervene earlier for maximum health of the crops and yield.

2.1.1. DEMERITS

1. Overfitting with Small Data: Transfer Learning is based on pre-trained models that have already been trained on large and varied datasets. While fine-tuning on a small dataset such as yours, overfitting may occur because there is not enough data to properly generalize the model to new examples.

2. Limited Generalization: The pre-trained model may not accurately capture the precise nuances of rice leaf diseases. It may not generalize to your dataset's precise variations or newly occurring diseases that have not been addressed in the initial pre-trained model.

3. Relying on Pre-trained Models: Transfer Learning frequently relies highly on the efficacy and appropriateness of the pre-trained model. If the pre-trained model fails to largely represent the characteristics of rice leaf diseases, then it may impede the improved performance obtained from transfer learning.

4. Domain Shift and Bias in the Dataset: If the dataset gathered from rice fields and the internet has biases or is not universally representative of all potential variations and situations that arise in actual rice farming, the model may have difficulty with actual usage and varied environmental conditions.

2.2. PROPOSED SYSTEM

The envisaged system, "Rice Leaf Disease Classification," utilizes the Mobile Net algorithm for effective and precise classification of rice leaf disease. Utilizing Mobile Net light architecture, the system facilitates processing in real time on mobiles. This minimizes accessibility problems for farmers by facilitating timely diagnosis and intervention against diseases. Implementation of advanced machine learning algorithms strengthens the system with the capability to assist precision farming, enabling good crop management, and achieving enhanced yields in paddy cultivation.

2.2.1. MERITS

1. Efficient Feature Extraction: Transfer Learning takes advantage of pre-trained models such as VGG16 or ResNet-50, trained on vast datasets such as ImageNet. These models have acquired rich representations of different visual features that can prove to be extremely useful in detecting patterns in rice leaf disease images. Utilizing these pre-trained features can greatly increase the accuracy of disease classification.

2. Small Dataset Usage: Since there is a small dataset, Transfer Learning plays a significant role. The early layers of pre-trained models such as VGG16 or ResNet-50 are used as feature extractors, allowing the model to understand certain features pertaining to rice leaf diseases using a small dataset. This method reduces the necessity of a large dataset for training.

3. Minimized Training Time: Training a CNN from scratch on a small dataset may result in overfitting and inefficiency. Transfer

Learning enables the fine-tuning of pre-trained models by modifying only certain layers, which minimizes training time by a large margin while improving or keeping the performance of the model intact.

4. Enhanced Generalization and Accuracy: With Transfer Learning, the model is able to generalize more effectively on unseen data. Pre-trained features learned by the pre-trained model tend to be general and transferable to a large set of visual recognition tasks, leading to enhanced accuracy in rice leaf disease classification.

2.3. FEASIBILITY STUDY

2.3.1. TECHNICAL FEASIBILITY

This experiment is executed to test the technical feasibility, i.e., the technical needs of the system. Any system developed should not have high demands on the available technical resources. This will result in high demands on the available technical resources. This will result in high demands on the client. The system that will be developed should have a moderate requirement, as very minimal or zero changes are needed to implement this system.

2.3.2. ECONOMICAL FEASIBILITY

This research is conducted to test the financial impact that the system will make in the company. The fund amount that can be invested by the company in research and development of the system is limited. The spending needs to be justified. Therefore, the system developed as well as in budget and that was made possible because most of the used technology is free of cost. The only customized items needed to be bought.

2.3.3. SOCIAL FEASIBILITY

The area of research is to verify the acceptance level of the system by the user. It involves the act of training the user in order to use the system effectively. The user should not fear the system but, rather, have to accept it as something indispensable. The degree of acceptance by the users is entirely based on the approach that is used to train the user regarding the system and to acquaint him with it. His confidence level should be increased so that he can also provide some constructive criticism, which is appreciated, as he is the end user of the system.

CHAPTER 3

SYSTEM DEVELOPMENT MODEL

The project adheres to a systematic, iterative development process with aspects of the Waterfall Model (for unambiguous phase-wise advancement) and Agile Methodology (for adaptability in testing and improvement). Below is the segmentation:

1. Requirement Analysis (Waterfall Phase)

Objective: Establish project objectives, scope, and limitations.

Activities:

- Collect farmer requirements (early disease detection).
- Determine technical requirements (Python, TensorFlow, Flask).
- Acquire dataset (Kaggle + field images).
- Output: SRS (Software Requirements Specification) document.

2. System Design (Waterfall Phase)

Architecture:

- Frontend: Web application using Flask (HTML/CSS/JS).
- Backend: CNN model (VGG16/MobileNet) + Transfer Learning.
- Database: MySQL to store user credentials.
- Tools: PyCharm IDE, TensorFlow/Keras libraries.

3. Prototype Development (Agile Iteration 1)

Target: Develop a minimal viable product (MVP).

Steps:

- Dataset preprocessing (resizing, normalization).
- Train a baseline CNN model (e.g., VGG16).
- Implement a simple Flask UI for uploading images.

- Validation: Test accuracy on a subset of the dataset.

4. Implementation & Testing (Agile Iterations 2-N)

Iterative Improvements:

- Model Enhancement: Refine with MobileNet for mobile support.
- UI Refinement: Include user registration/login (MySQL integration).
- Performance Testing: Test accuracy, latency, and scalability.

Testing Types:

- Unit Testing: Test individual modules (e.g., CNN layers).
- Integration Testing: Flask + CNN model interaction.
- User Acceptance Testing (UAT): Farmers use the deployed app.

5. Deployment (Waterfall + Agile)

Steps:

- Deploy Flask app on cloud platforms (AWS/Heroku).
- Optimize model for edge devices (farmers' smartphones).
- Monitoring: Measure actual usage and accuracy drift.

6. Maintenance & Updates (Continuous Agile)

Activities:

- Retrain model with fresh disease data.
- Bugs fix (e.g., image upload issue).
- Features add (multilingual support, SMS notification).
- Selected Model Rational

CHAPTER 4

SYSTEM DESCRIPTION

4.1. PROBLEM DEFINITION

Rice is among the world's most vital staple crops and a main food source for more than three billion people. Rice plants are prone to several diseases at various phases of cultivation, which can dramatically affect yield and quality. It is common for farmers to fail to correctly diagnose these diseases owing to inadequate knowledge and resources. Conventional disease detection methods are entirely dependent on manual checking, which is time-consuming and subject to human error. Moreover, the availability of agricultural specialists is restricted, particularly in rural regions, rendering timely and correct diagnosis even more difficult. Such drawbacks indicate that there is an indispensable need for an automated, efficient, and accessible measure for early rice disease detection.

Recent developments in machine learning and computer vision present promising solutions to this issue. Although conventional machine learning methods such as Support Vector Machines (SVM) and Artificial Neural Networks (ANN) have been considered, their use is usually thwarted by the requirement for hand-crafted feature extraction and the reliance on huge, high-quality datasets. Convolutional Neural Networks (CNNs) have proven to be a better option, where features can be automatically extracted from images without significant preprocessing. Still, training CNNs from the beginning needs lots of labeled data, which usually is not present for particular applications in agriculture, such as the detection of rice diseases. Such a lack of data is one of the most important obstacles in creating robust and accurate models.

To overcome these challenges, this project suggests an automated system using Transfer Learning with pre-trained CNN models such as VGG16 and MobileNet. Transfer Learning enables the utilization of models trained on large general datasets (e.g., ImageNet) for specific tasks, like rice disease classification, even with small data. Fine-tuning these models by the system enables it to attain high accuracy while avoiding the requirement of large datasets. In addition, the system is mobile platform deployable, allowing farmers to send pictures of infected leaves through smartphones with instant diagnosis and treatment suggestions. This not only enhances accessibility but also facilitates timely intervention to prevent crop loss.

The system suggested here seeks to outdo traditional approaches by harnessing the capability of CNNs along with farmer-friendly design that is practical. Important goals are the high classification rate for prevalent diseases in rice, the ability to scale up to be used with real-world conditions, and to be able to provide actionable data for farmers. Closing the divide between sophisticated AI technology and crop requirements, the system can help revolutionize rice cultivation, raise productivity, save costs, and advance global food security. The end result is to equip farmers with a scientifically sound and simple-to-use tool that changes the face of rice disease detection and management.

4.2. OVERVIEW OF THE SYSTEM

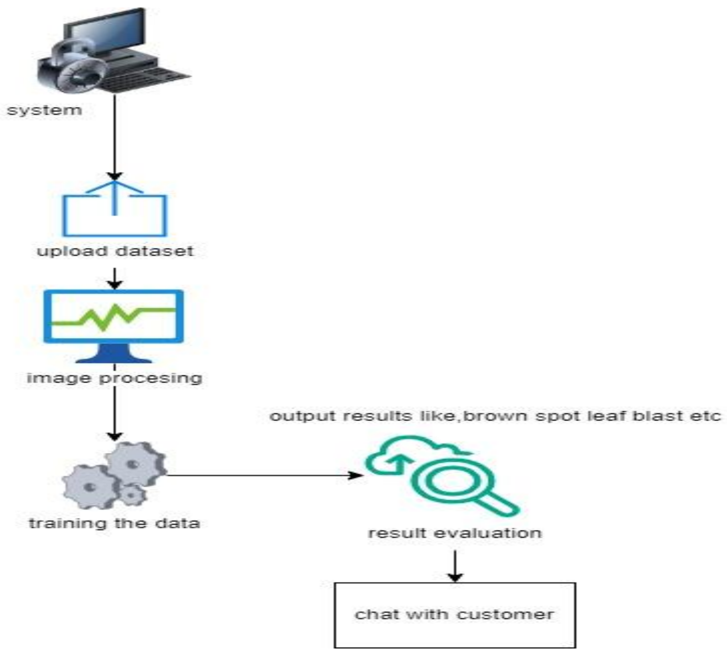
The suggested automated rice leaf disease diagnosis system is intended to present farmers with a precise, effective, and easy-to-use solution in diagnosing prevalent rice diseases. Fundamentally, the system employs the use of deep learning technology, namely Convolutional Neural Networks (CNNs) with transfer learning, to examine photographs of rice leaves and identify possible diseases. The architecture of the system is comprised of three primary elements: a web or mobile frontend interface for farmers to upload images, a backend server for analysis and processing, and a pre-trained CNN model that has been fine-tuned on a custom dataset of rice leaf images. This combined strategy allows even farmers with minimal technical skills to access and utilize sophisticated disease detection capabilities with ease.

The frontend application is the main interface between the farmers and the system. It has been designed to be simple so that users can take or upload pictures of suspected diseased rice leaves using their smartphones or computers. The application is optimized to run on low-end devices and in low internet connectivity areas to ensure wide accessibility. Once an image has been uploaded, the system provides clear output, such as the name of the disease identified, the level of confidence in diagnosis, and pragmatic treatment advice. This feedback loop is instant, which empowers farmers to make informed action on their crops in time.

On the backend side, the system processes images and classifies the diseases. When a farmer uploads an image, the server preprocesses the image initially by resizing and normalizing it to satisfy the input requirements of the CNN model. The system then employs the fine-tuned model to analyze the image and predict whether there are diseases like Leaf Blast, Bacterial Blight, or Brown Spot. As a means of offsetting sparse training data, the system utilizes data augmentation methods, synthetically enlarging the data through variations of already existing images using rotations, zooms, and flips. The backend also has a database of disease facts and solutions, allowing it to give farmers advice based on the diagnosis.

The deep learning model constitutes the smarts of the system. Taking advantage of transfer learning, the system is based on pre-trained CNN models such as VGG16 or MobileNet, initially trained on massive image datasets. By fine-tuning these models on a niche dataset of rice leaf images, the system performs well in disease detection with relatively small amounts of training data. The model is performance-optimized to enable rapid processing even on mobile devices. This blend of cutting-edge technology and sensible design renders the system a formidable instrument for enhancing crop health management and aiding farmers in making responsive decisions in order to boost rice production.

4.3. SYSTEM ARCHITECTURE DIAGRAM



CHAPTER 5

SYSTEM DESIGN

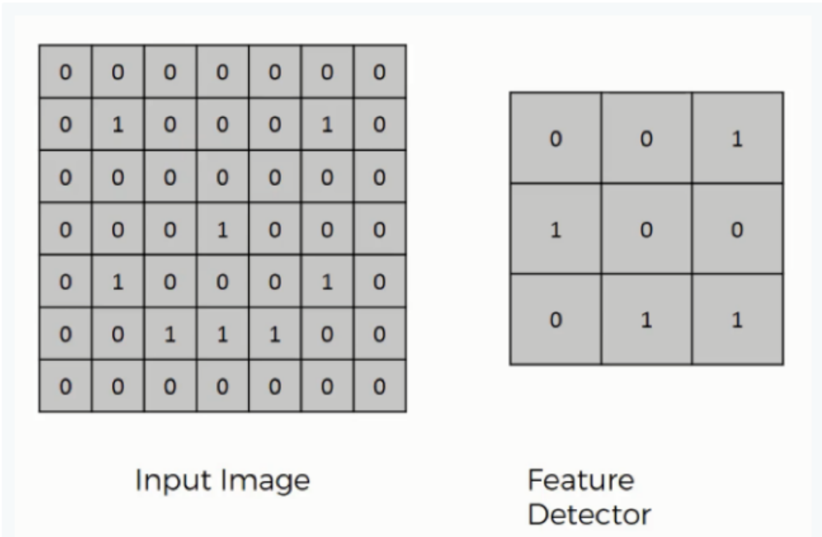
5.1. MODULE AND IT'S DESCRIPTION AND ALGORITHM

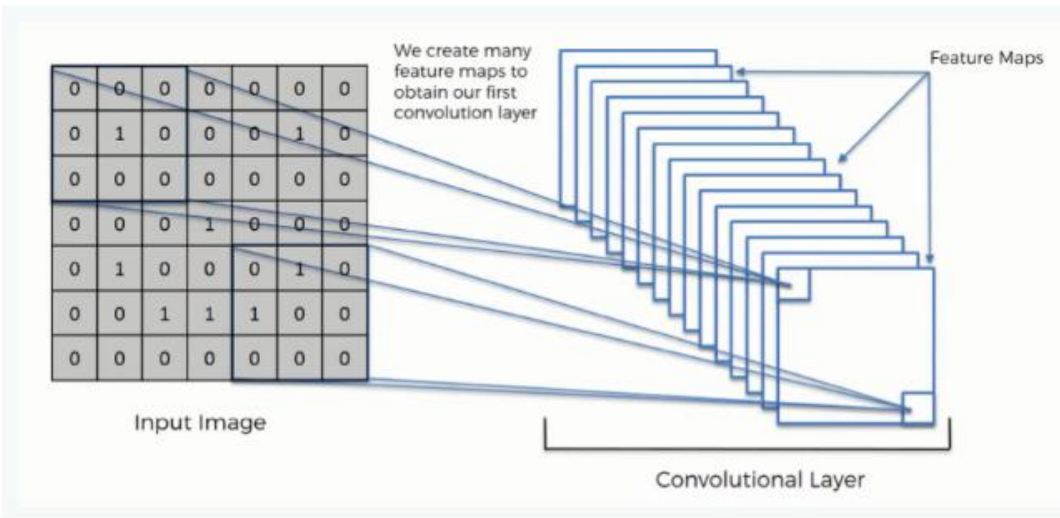
Convolutional Neural Network

Step1: convolutional operation

The initial building step in our attack plan is the operation of convolution. This is where we hit feature detectors, essentially the filters for the neural network. We will address feature maps, learning the parameters of the said maps, how patterns are detected, the detection layers, and how the results are mapped.

The Convolution Operation



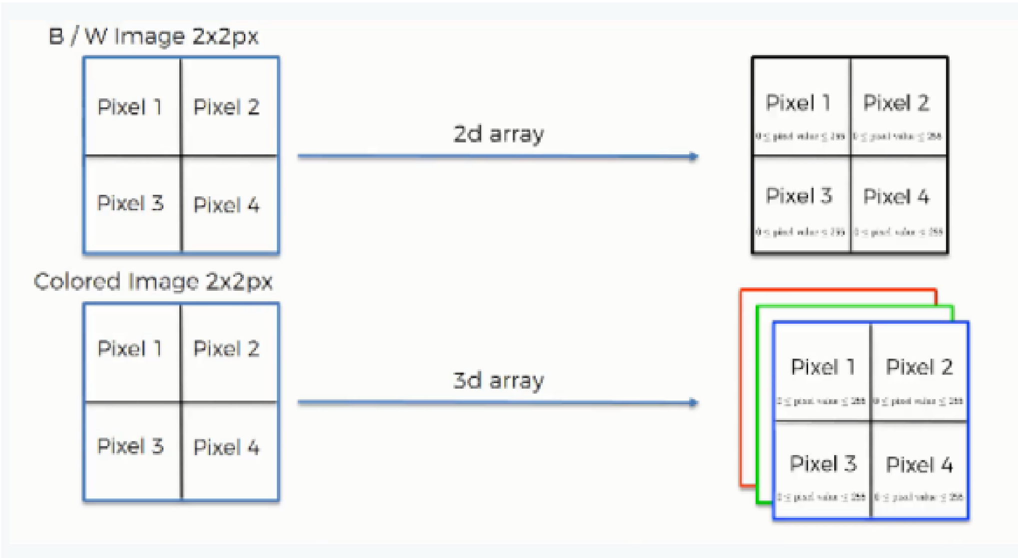


Step (1b): Relu Layer

The second half of this step will be the Rectified Linear Unit or Relook. We will discuss Relook layers and how linearity operates within Convolutional Neural Networks.

Not required to learn CNN's, but there's no harm in a quick introduction to enhance your proficiency.

Convolutional Neural Networks Scan Images



Step 2: Pooling Layer

Here, we'll discuss pooling and will come to know precisely how it typically operates. Our focus point here, though, will be a particular kind of pooling; max pooling. We'll discuss several methods, though, such as mean (or sum) pooling. This section will conclude with an example created with a visual interactive tool that will certainly get the entire idea out for you.

Step 3: Flattening

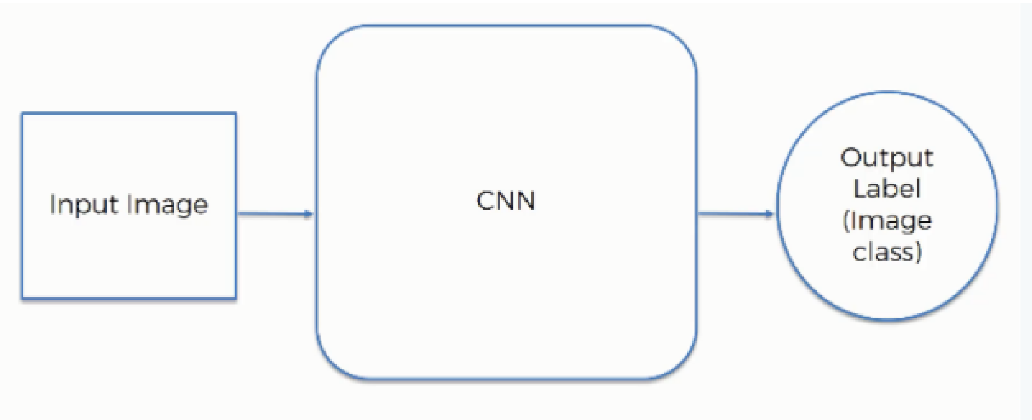
This shall be a step-by-step summary of the process of flattening and how we transition from pooled to flattened layers when dealing with Convolutional Neural Networks.

Step 4: Full Connection

In this section, all that we have discussed till now will be combined. With this, you'll be able to visualize a better idea of how Convolutional Neural Networks work and how the "neurons" which are ultimately generated learn to classify images.

Summary

Finally, we will summarize everything and have a brief overview of the topic discussed in the section. If you want to, it will do you some good (and it likely will), you should look at the additional tutorial where Soft ax and Cross-Entropy are discussed. It's not required for the course, but you will probably encounter these when using Convolutional Neural Networks and it will do you a lot of good to know them.

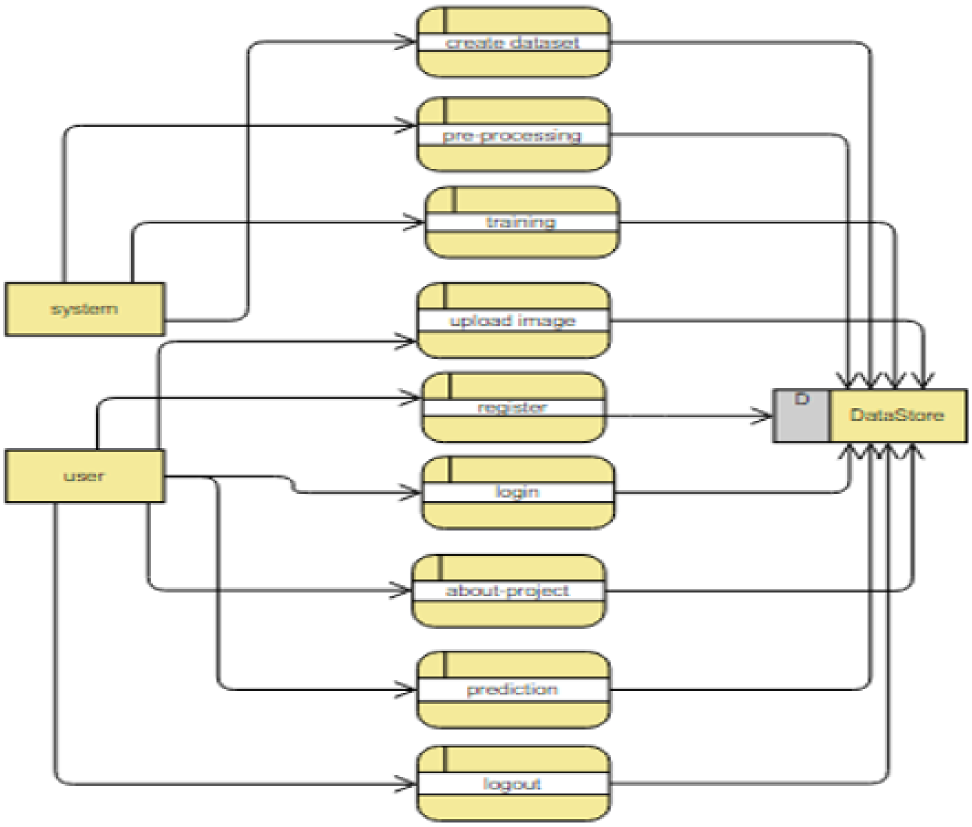


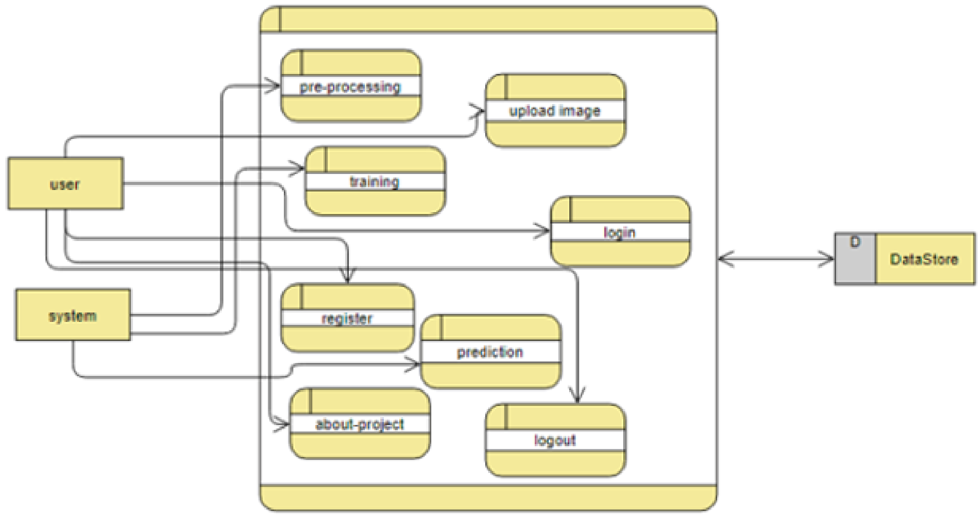
MOBILENET

MobileNet is a light-weight convolutional neural network (CNN) architecture for accurate and efficient image classification tasks and is applicable in rice leaf disease classification, among other uses. Its major innovation is depthwise separable convolutions, which greatly minimize computational complexity while retaining high performance. MobileNet's efficient architecture makes it highly appropriate for deployment on mobile devices with limited computational power, making it a suitable candidate for real-time rice leaf disease classification. Its trade-off between accuracy and efficiency makes it a widely used candidate for mobile-based machine learning tasks, ensuring effective disease diagnosis and management in the case of rice crops.

5.2. DFD DIAGRAM

A Data Flow Diagram (DFD) is an old method of illustrating the information flows in a system. A clean and concise DFD can illustrate a considerable portion of the system requirements pictorially. It may be manual, automatic, or semi-automatic. It illustrates how information flows into and out of the system, what transforms the information and where information is stored. The intention of a DFD is to illustrate the bounds and scope of a system overall. It can be employed as a communication vehicle between a systems analyst and anyone who has some role in the system that is being used as the beginning of redesigning a system.





5.3. UML DIAGRAM

UML refers to Unified Modeling Language. UML is an official standard, and it's a general-purpose modeling language of the object-oriented software engineering domain. Management, and its formulation by, of the standard belongs to the Object Management Group.

The aim is for UML to be an everyday language used to develop models of object oriented computer programs. As it stands today UML consists of two large parts: a Meta-model and a notation. In the future, a method or process of some kind may also be included in; or linked to, UML.

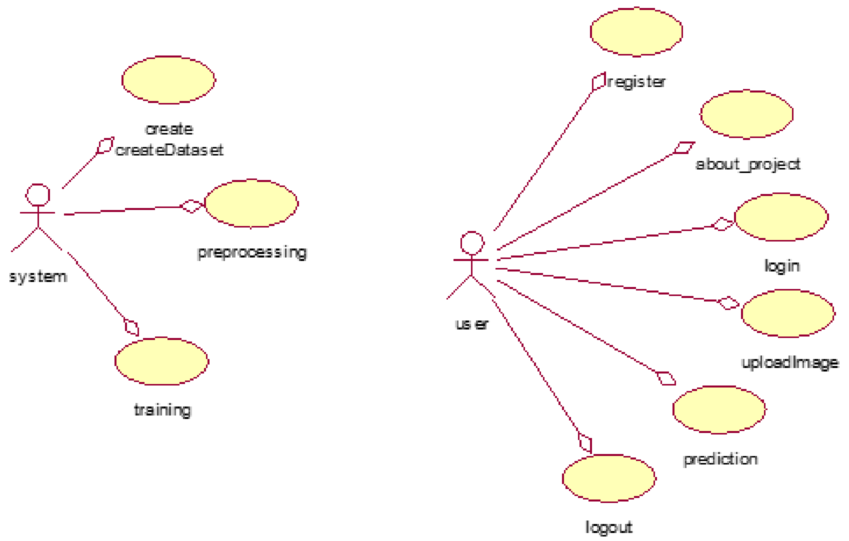
Unified Modeling Language is a standard language for business modeling and specifying, Visualization, Constructing and documenting the software system's artifacts, as well as other non-software systems.

The UML is a set of best engineering practices that have been successful in modeling large and complex systems.

The UML constitutes an extremely significant component of object oriented software development and the software development process. The UML employs primarily graphical notations to represent the software project's design.

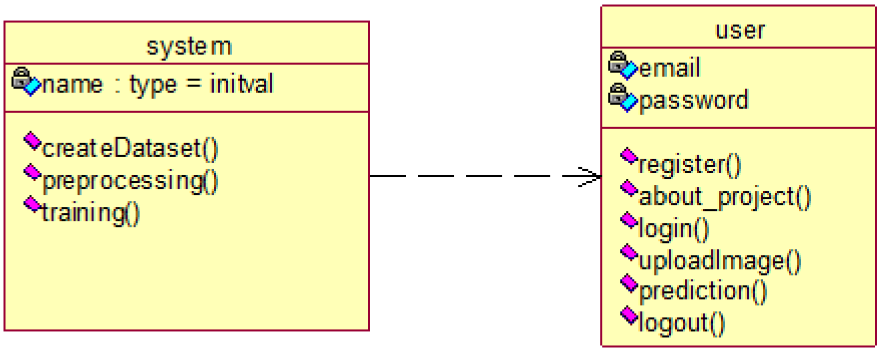
USE CASE DIAGRAM:

An use case diagram within the Unified Modeling Language (UML) is a form of behavioral diagram stipulated by and derived from a Use-case analysis. It aims to express in a graphical view the system functionality in terms of actors, actors' intentions (in the guise of use cases), and how these use cases are interdependent. The primary goal of a use case diagram is to indicate which system functions are executed on behalf of which actor. Actor roles within the system can be illustrated.



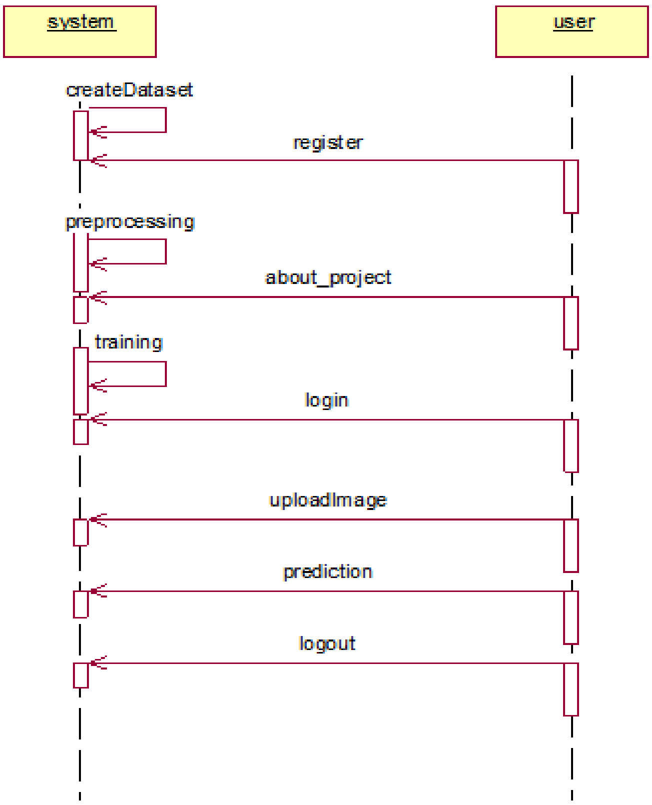
CLASS DIAGRAM:

In software development, a class diagram in the Unified Modeling Language (UML) is a kind of static structure diagram that illustrates the structure of a system by indicating the classes of the system, their attributes, operations (or methods), and the relationships between the classes. It clarifies which class holds information.



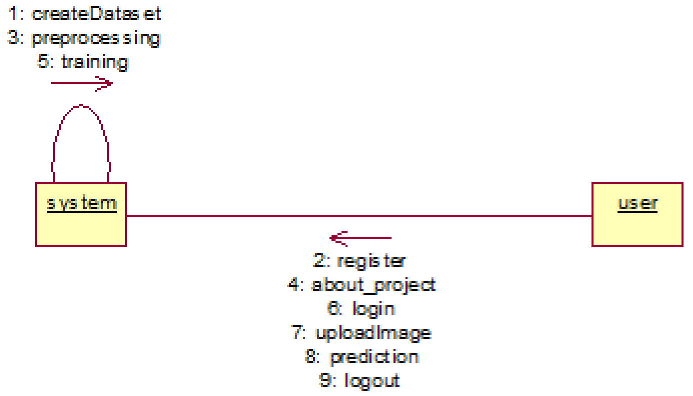
SEQUENCE DIAGRAM:

A Unified Modeling Language (UML) sequence diagram is a type of interaction diagram that illustrates how processes interact with each other and in what sequence. It is an abstraction of a Message Sequence Chart. Sequence diagrams are also referred to as event diagrams, event scenarios, and timing diagrams.



Collaboration Diagram:

In collaboration diagram the sequence of method calls is represented by some sort of numbering method as depicted below. The number defines the way methods are invoked one after another. We have used the same order management system to explain the collaboration diagram. The method calls are like that of a sequence diagram. But the difference lies in the fact that sequence diagram does not specify the object organization while the collaboration diagram depicts the object organization.

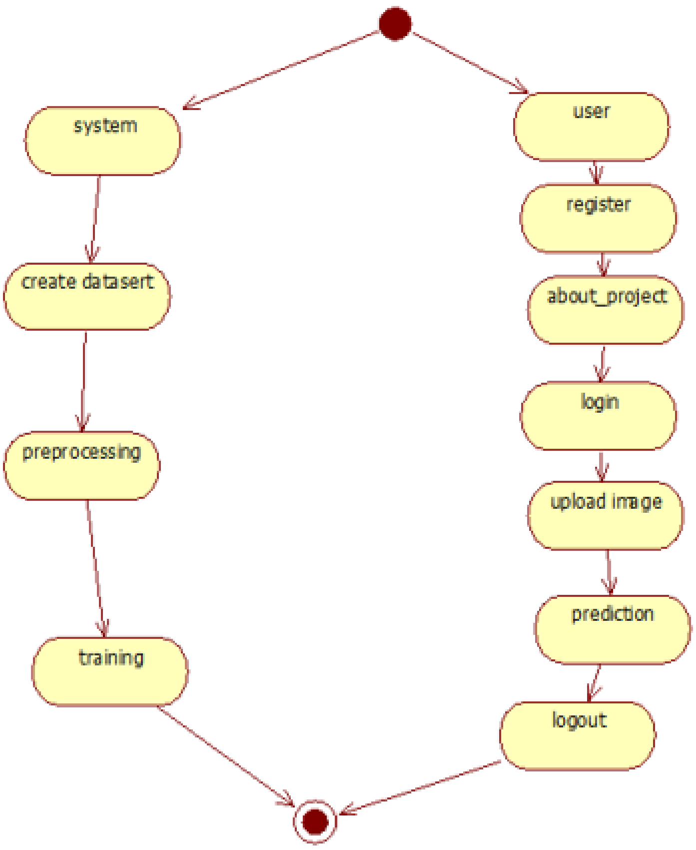


DEPLOYMENT DIAGRAM

Deployment diagram depicts the deployment perspective of a system. It has a relation with the component diagram. Since the components are deployed by means of the deployment diagrams. A deployment diagram comprises nodes. Nodes are nothing but physical hardware's to deploy the application.

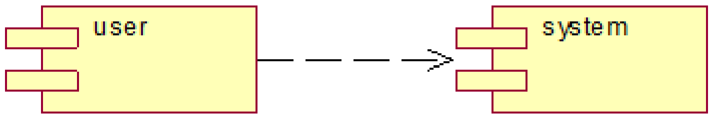
ACTIVITY DIAGRAM:

Activity diagrams are stepwise workflow graphs of stepwise activities and actions with choice, iteration, and concurrency support. Activity diagrams in the Unified Modeling Language can be applied to describe the operational and business step-by-step workflows of components within a system. An activity diagram illustrates the total control flow.



COMPONENT DIAGRAM:

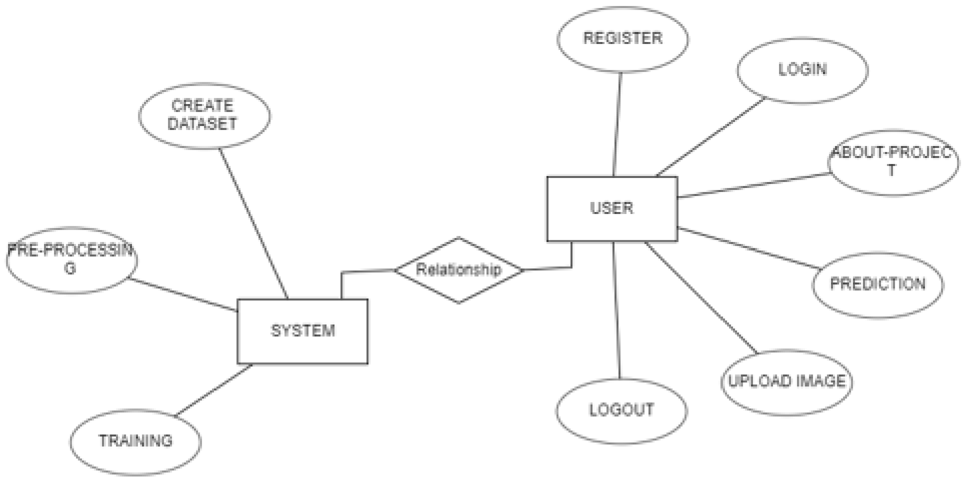
A component diagram, or UML component diagram, specifies the wire-up and organization of physical components within a system. Component diagrams are frequently created to assist in modeling implementation concerns and ensure that all facets of the required functions of the system are addressed by development planning.



5.4. ER DIAGRAM

An Entity–relationship model (ER model) defines the organization of a database using a diagram, which is referred to as Entity Relationship Diagram (ER Diagram). An ER model is a design or a plan of a database that can subsequently be realized as a database. The primary elements of E-R model are: entity set and relationship set.

An ER diagram displays the relationship between entity sets. An entity set is a collection of similar entities and these entities may have attributes. In DBMS, an entity is a table or attribute of a table in a database, so by displaying relationship between tables and their attributes, ER diagram displays the entire logical structure of a database. Let's see an example of simple ER diagram to see how this is actually done.



5.3. Input Design:

Input in an information system is raw data that is processed in order to generate output. While designing the input, the developers need to take into consideration the input devices like PC, MICR, OMR, etc.

Hence, the quality of system input will decide the quality of system output. Properly designed input forms and screens possess the following attributes −

- It should be effective to serve specific purpose like storing, recording, and retrieving the information.
- It guarantees proper completion with precision.
- It must be simple to fill and easy.
- It must be user's attention, consistent, and simple.
- All these goals are achieved through the understanding of fundamental design principles concerning −
  - What are the inputs required by the system?
  - How end users react to various elements of forms and screens.

Objectives for Input Design:

The goals of input design are −

- To design data entry and input processes
- To minimize input quantity
- To create source documents to capture data or other data capture techniques
- To create input data records, data entry screens, user interface screens, etc.
- To implement validation checks and good input controls.

5.4. Output Design:

The output design is the most critical activity of any system. In output design, developers determine the kind of outputs that would be required, and take into consideration the output controls and prototype report formats that would be required.

Objectives of Output Design

The goals of input design are:

- To create output design that performs the purpose and eliminates unwanted production of output.
- To create the output design that satisfies the end user's needs.
- To provide the right amount of output.
- To shape the output in proper form and direct it to the correct individual.
- To provide the output on time to make good decisions.

CHAPTER 6

SYSTEM SPECIFICATION

6.1. HARDWARE CONFIGURATION

Processor	-	I3/Intel Processor
Hard Disk	-	160 GB
Key Board	-	Standard Windows Keyboard
Mouse	-	Two or Three Button Mouse
Monitor	-	SVGA
RAM	-	8 GB

6.2. SOFTWARE CONFIGURATION

Operating System	-	Windows 7/8/10
IDE	-	PyCharm
Libraries Used	-	Numpy,IO,OS,Flask,Keras
Technology Used	-	Python 3.6+

CHAPTER 7
SYSTEM TESTING

7.1. DEFINITION

System testing is a thorough testing phase during which the entire software system is tested to ensure that it satisfies the given requirements and behaves as required. It is performed on a fully integrated system to test the end-to-end functioning and overall performance of the application. This type of testing is commonly the last step prior to deployment and includes testing all functional and non-functional dimensions of the system. In contrast to unit or integration testing, system testing considers the software as a "black box" and only deals with inputs and outputs without looking at how the code works from the inside.

The objective of system testing is to:

1. Verify that all the components work together properly as an integrated system
2. Determine whether there are any missing requirements or functionality
3. Test the user interface, workflow, and data flow
4. Ensure the software works reliably under different usage scenarios

7.2. LEVELS OF TESTING

The testing levels are the various phases where software testing is done to check quality and functionality. Below are the main levels of testing, classified according to their scope and objective:

1. Unit Testing

Objective: To test single units of the software independently (e.g., functions, methods, or classes).

Carried out by: Developers.

Main Focus:

Internal code structure and logic.

Checking inputs and outputs.

Ensuring zero defects in the smallest testable units.

2. Integration Testing

Purpose: To check whether various modules or services function as desired when they are integrated.

Types:

Top-down Integration (Testing top-level modules first, with stubs for the lower ones).

Bottom-up Integration (Testing lower-level modules first, with drivers for the top ones).

Sandwich/Hybrid Integration (Both top-down and bottom-up combined).

Key Focus:

Interface faults.

Data passing between modules.

Interaction between the integrated parts.

3. System Testing

Purpose: To test the entire and fully integrated software system against defined requirements.

Includes:

Functional Testing (Validating system functions).

Non-functional Testing (Performance, security, usability, etc.).

Performed by: Independent testing team.

Key Focus:

- End-to-end system behavior.
- Compliance with business and technical requirements.

4. Acceptance Testing

Purpose: To decide whether the system is production-ready and fulfills business/user requirements.

Types:

User Acceptance Testing (UAT) – Performed by end-users.

Business Acceptance Testing (BAT) – Validates business processes functioning.

Alpha/Beta Testing – Alpha (in-house), Beta (external users).

Key Focus:

Verifying real-world scenarios.

Making sure the system is suitable for business use.

7.3. Test case design technique

Test case design methods are techniques applied to design effective test cases to achieve maximum test coverage with the least redundancy. They efficiently identify defects and fall under the category of Black-Box (functional) and White-Box (structural) techniques.

1.Black Box Testing Techniques

2.White Box Testing Techniques

7.3.1. Test Strategy and Approach

Field testing shall be done by hand and functional tests shall be written out thoroughly.

Test purposes

- All field inputs should function fine.
- Pages should be triggered from the found link.
- The entry screen, messages and responses should not be sluggish.

Features to be tested

- Confirm that the entries are in the right format
- There should be no duplicate entries allowed
- all links should take the user to the right page.

Integration Testing

Software integration testing is incremental integration testing of two or more integrated software components on a single platform to generate failures due to interface defects.

The integration test's job is to verify that components or software applications, e.g. components in a software system or – one level up – software applications at the company level – communicate without error.

Test Results: All the above-mentioned test cases passed successfully. No defects were encountered.

Acceptance Testing

User Acceptance Testing is a very important part of any project and involves major involvement by the end user. It also checks whether the system is meeting the functional requirements or not.

Test Results: All the above-mentioned test cases passed successfully. No defects were encountered.

CHAPTER 8

SYSTEM IMPLEMENTATION

8.1 Modules

Implementation of the Rice Leaf Disease Classification System includes the following modules:

1. Dataset Creation & Preprocessing

Dataset Collection:

Images gathered from rice fields in West Bengal (India) and online.

Total dataset: 1,509 training images (Rice Blast, Leaf Blight, Brown Spot, Healthy) + 647 test images.

Data Augmentation:

Techniques used: rotation, zoom, horizontal/vertical shifts (using Keras ImageDataGenerator).

Aids to avoid overfitting due to small dataset.

Preprocessing:

Resizing images to 224x224 pixels (VGG16 input size).

Normalization (pixel values / 255).

2. Model Training (Transfer Learning with VGG16)

- **Base Model**: **VGG16** (pre-trained on ImageNet, weights frozen).
- **Fine-Tuning**:
  - Added custom layers:

```python
from keras.applications import VGG16
from keras.layers import Dense, Flatten, Dropout
from keras.models import Model

base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
for layer in base_model.layers[:15]:  # Freeze initial layers
    layer.trainable = False

x = Flatten()(base_model.output)
x = Dense(256, activation='relu')(x)
x = Dropout(0.5)(x)  # Reduce overfitting
predictions = Dense(4, activation='softmax')(x)  # 4 classes
model = Model(inputs=base_model.input, outputs=predictions)
```

- **Training Parameters**:
  - Optimizer: **Adam** (lr=0.0001).
  - Loss: **Categorical Crossentropy**.
  - Epochs: **25** (early stopping applied).
  - Batch Size: **32**.

3. **Web Application (Flask Backend)**
   - **User Interface**:
     - **Upload Page**: Farmers upload leaf images via a web form.
     - **Prediction Page**: Displays disease diagnosis + remedy suggestions.
4. **Mobile Integration (Future Scope)**
   - **Android App**:
     - Uses **TensorFlow Lite** for on-device inference.
     - Farmers capture leaf images via phone camera → instant diagnosis.

ALGORITHM AND METHODOLOGY

Convolutional Neural Network (CNN) Pipeline

Input Layer:

Processes 224x224x3 RGB images.

Feature Extraction (VGG16):

Utilizes pre-trained convolutional layers to identify edges, textures, and patterns.

Fine-Tuned Layers:

Flatten: Flattens 3D feature maps to 1D vector.

Dense (256 neurons): Identifies disease-specific features.

Dropout (0.5): Avoids overfitting.

Output Layer (Softmax):

Produces probabilities for 4 classes (Rice Blast, Leaf Blight, Brown Spot, Healthy).

Training Workflow

Data Splitting:

80% Training (1,509 images).

20% Validation (for hyperparameter tuning).

Augmentation:

```
from keras.preprocessing.image import ImageDataGenerator
train_datagen = ImageDataGenerator(rotation_range=20,
zoom_range=0.2, horizontal_flip=True)
train_generator = train_datagen.flow_from_directory('train_dir',
target_size=(224, 224), batch_size=32)
```

Model Training:

- Monitored **validation accuracy** for early stopping.

Result and Performance

Metric	Value
Test Accuracy	92.46%
Precision	91.8%
Recall	92.1%
F1-Score	91.9%

Confusion Matrix Example:

	Blast	**Blight**	**Brown Spot**	**Healthy**
Blast	142	5	3	0
Blight	4	138	2	1
Brown Spot	2	4	145	0
Healthy	0	1	0	155

Challenges and Solution

Challenge	**Solution**
Small dataset (1,509 images)	Used **Transfer Learning** + **Data Augmentation**.
Overfitting	Added **Dropout (0.5)** and **Early Stopping**.
Slow training on CPU	Switched to **Google Colab (GPU)**.

CONCLUSION

In this paper we have suggested a deep learning model with training on 1509 rice leaf images and testing on various 647 images and correctly classifying 92.46% of the test images. Transfer Learning with fine-tuning the pre-defined VGG16 has significantly enhanced the performance of the model which otherwise was not yielding good results on such small dataset. The number of epochs utilized was brought to a halt at 25 since we had been given a cut point from which the accuracy was not increasing and the loss was not going down on training as well as validation data.

FUTURE ENHANCEMENT

In the future, we would like to gather more images from agricultural farms and Agricultural Research centers so that we can further enhance the accuracy. We would like to incorporate cross-validation process in the future to verify our results. We would also like to implement improved deep learning models and other state-of the art works and compare it with the results we got. The model developed here can be applied in the future to identify other plant leaf diseases, which are valuable crops in India.

REFERENCE

[1] T. Gupta, ”Plant leaf disease analysis using image processing technique with modified SVM-CS classifier,” Int. J. Eng. Manag. Technol, no. 5, pp. 11-17, 2017.

[2] Y. Es-saady,T. El Massi,M. El Yassa,D. Mammass, and A. Benazoun, ”Automatic recognition of plant leaves diseases based on serial combination of two SVM classifiers,” International Conference on Electrical and Information Technologies (ICEIT) pp. 561-566, 2016.

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

[4] L. Liu and G. Zhou, “Extraction of the rice leaf disease image based on BP neural network,” International Conference on Computational Intelligence and Software Engineering ,pp. 1-3,2009.

[5] S. Arivazhagan and S.V. Ligi, “Mango Leaf Diseases Identification Using Convolutional Neural Network,” International Journal of Pure and Applied Mathematics, vol. 120,no. 6, pp. 11067-11079,2008.

[6] B. Liu,Y. Zhang, D. He and Y. Li, “Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks Symmetry,” X. X. & Suen, C. Y. A novel hybrid CNN–SVM classifier for recognizing handwritten digits. Pattern Recognition,vol. 45,pp. 1318–1325,2012.

[7] Y. Lu, S. Yi,N. Zeng,Y. Liu, and Y. Zhang, “Identification of Rice Diseases Using Deep Convolutional Neural Networks,” Neurocomputing, 267, pp. 378-384,2017.

[8] R. R. Atole, D. Park, “A Multiclass Deep Convolutional Neural Network Classifier for Detection of Common Rice Plant Anomalies,” International

Journal Of Advanced Computer Science And Applications,vol. 9,no. 1,pp. 67–70,2018.

[9] V. Singh, A. Misra, “Detection of Plant Leaf Diseases Using Image Segmentation and Soft Computing Techniques,” Information Processing in Agriculture,vol. 4 ,no. 1,pp. 41–49,2017.

[10] P. Konstantinos Ferentinos, “Deep Learning Models for Plant Disease Detection and Diagnosis,” Computers and Electronics in Agriculture ,vol. 145,pp. 311–318,2018.

[11] M. Brahimi, K. Boukhalfa, and A. Moussaoui, “Deep Learning for Tomato Diseases: Classification and Symptoms Visualization,” Applied Artificial Intelligence, vol. 31, no. 4, pp. 299–315, 2017.

[12] S. Mohanty, D. P. Hughes, and M. Salathé, “Using Deep Learning for Image-Based Plant Disease Detection,” Frontiers in Plant Science, vol. 7, p. 1419, 2016.

[13] P. Ghosal, S. Sarkar, and A. A. Das, “Rice Leaf Disease Detection Using Convolutional Neural Networks,” IEEE Region 10 Conference (TENCON), pp. 2561-2566, 2018.

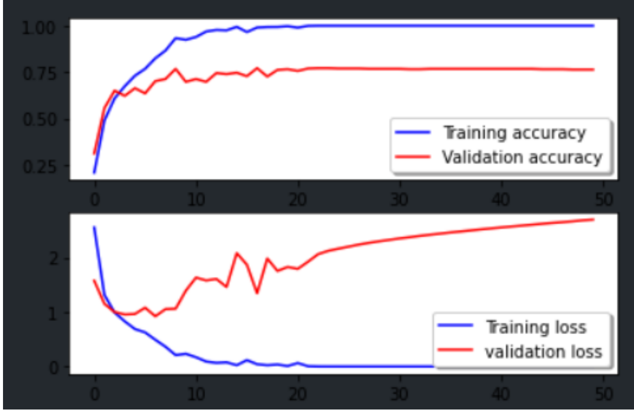
[14] R. P. Meena, S. Raja, and S. Arunprasath, “Identification of Rice Leaf Diseases Using Deep CNN Models,” International Journal of Advanced Research in Computer Science, vol. 9, no. 2, pp. 43–47, 2018.

[15] H. Too, L. Yujian, S. Njuki, and L. Yingchun, “A Comparative Study of Fine-Tuning Deep Learning Models for Plant Disease Identification,” Computers and Electronics in Agriculture, vol. 161, pp. 272–279, 2019.

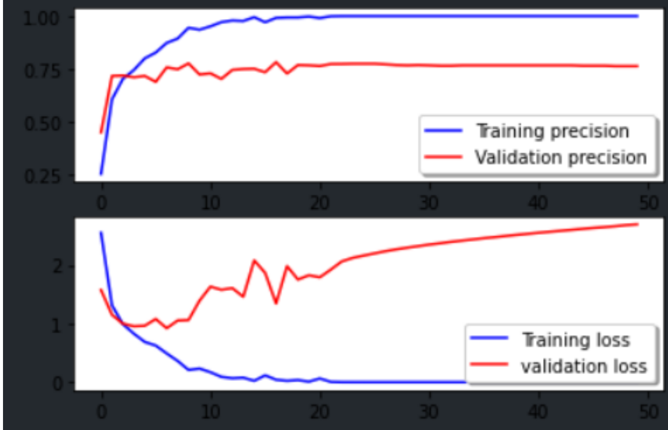
APPENDIX

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 256, 256, 32)      2432
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 256, 256, 32)      25632
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 128, 128, 32)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 128, 128, 64)      18496
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 128, 128, 64)      36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 64, 64, 64)        0
_________________________________________________________________
flatten (Flatten)            (None, 262144)            0
_________________________________________________________________
dense (Dense)                (None, 256)               67109120
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 1542
=================================================================
Total params: 67,194,150
Trainable params: 67,194,150
Non-trainable params: 0
_________________________________________________________________
```

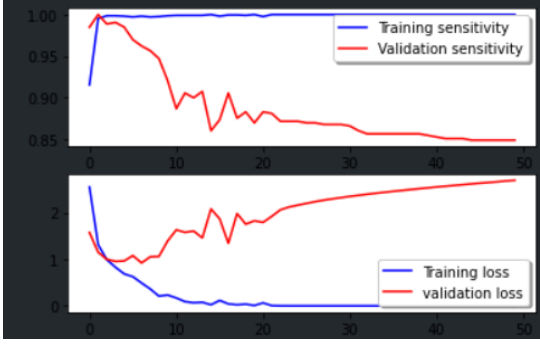
```
fig, ax = plt.subplots(2,1)
ax[0].plot(hist.history['accuracy'], color='b', label="Training accuracy")
ax[0].plot(hist.history['val_accuracy'], color='r',label="Validation accuracy")
legend = ax[0].legend(loc='best', shadow=True)

ax[1].plot(hist.history['loss'], color='b', label="Training loss")
ax[1].plot(hist.history['val_loss'], color='r', label="validation loss")
legend = ax[1].legend(loc='best', shadow=True)
```



```
fig, ax = plt.subplots(2,1)
ax[0].plot(hist.history['precision'], color='b', label="Training precision")
ax[0].plot(hist.history['val_precision'], color='r',label="Validation precision")
legend = ax[0].legend(loc='best', shadow=True)

ax[1].plot(hist.history['loss'], color='b', label="Training loss")
ax[1].plot(hist.history['val_loss'], color='r', label="validation loss")
legend = ax[1].legend(loc='best', shadow=True)
```

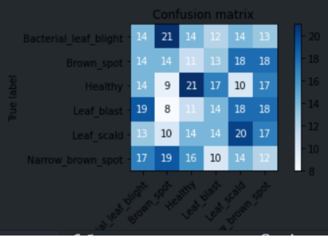



```
fig, ax = plt.subplots(2,1)
ax[0].plot(hist.history['sensitivity_at_specificity'], color='b', label="Training sensitivity")
ax[0].plot(hist.history['val_sensitivity_at_specificity'], color='r',label="Validation sensitivity")
legend = ax[0].legend(loc='best', shadow=True)

ax[1].plot(hist.history['loss'], color='b', label="Training loss")
ax[1].plot(hist.history['val_loss'], color='r', label="validation loss")
legend = ax[1].legend(loc='best', shadow=True)
```

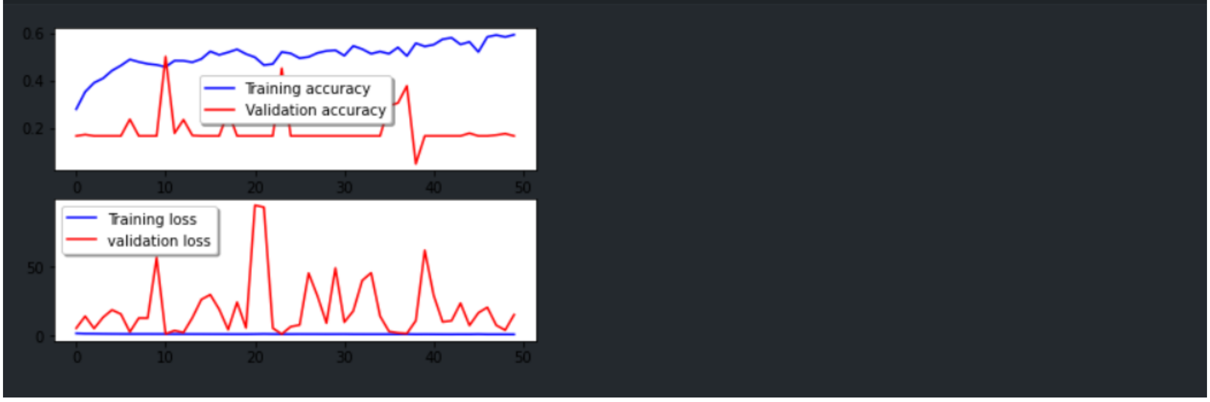


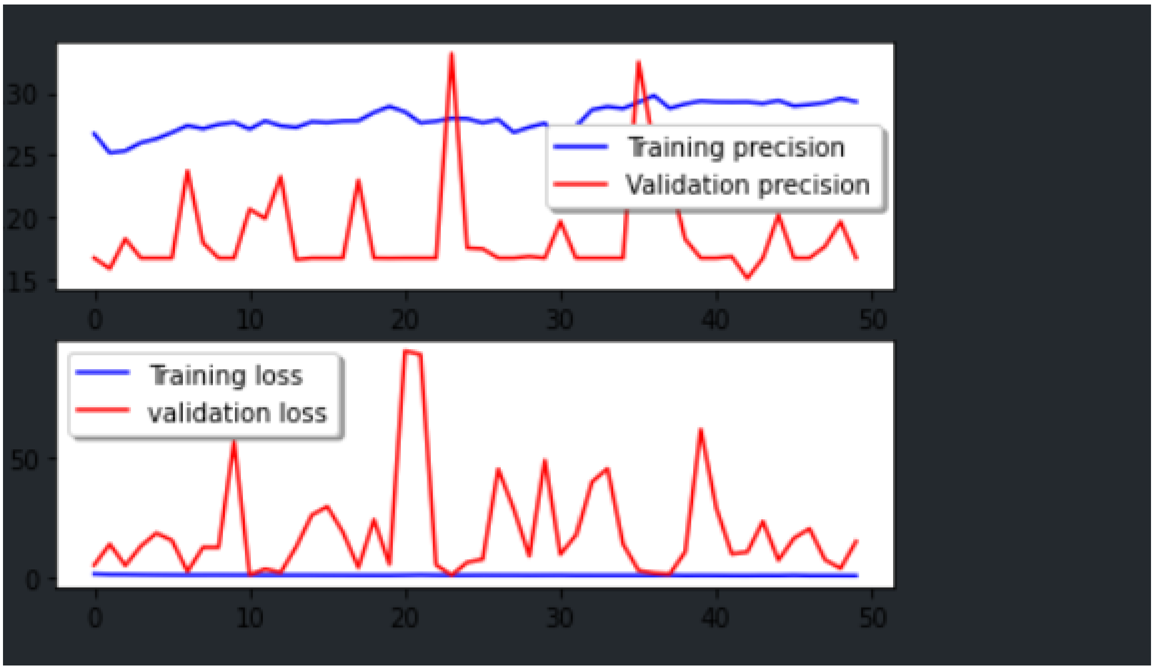
```python
y_true = test_generator.classes
y_pred = model_predict.argmax(axis=1)
plot_confusion_matrix(y_true, y_pred, classes=['Bacterial_leaf_blight', 'Brown_spot','Healthy','Leaf_blast','Leaf_scald','Narrow_brown_spot'],
                      title='Confusion matrix')
```

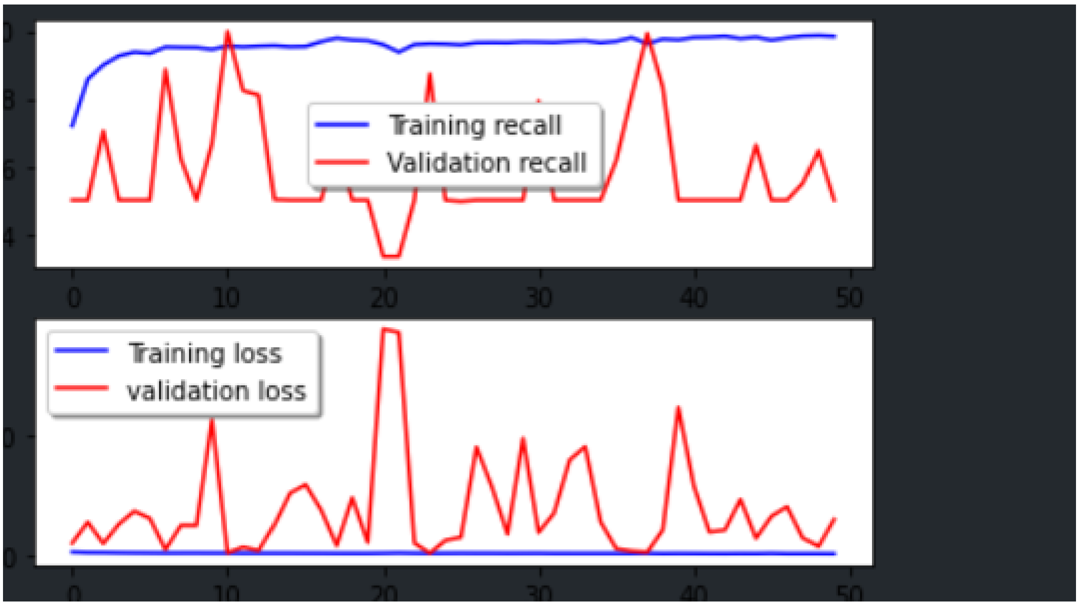
```
Confusion matrix, without normalization
[[14 21 14 12 14 13]
 [14 14 11 13 18 18]
 [14  9 21 17 10 17]
 [19  8 11 14 18 18]
 [13 10 14 14 20 17]
 [17 19 16 10 14 12]]

<AxesSubplot:title={'center':'Confusion matrix'}, xlabel='Predicted label', ylabel='True label'>
```

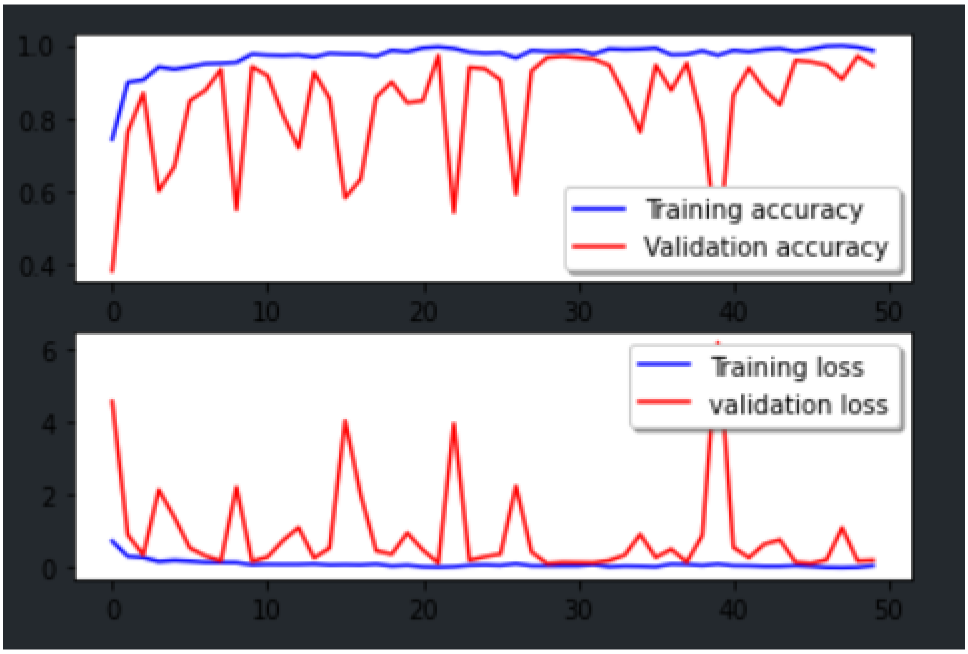


```
fig, ax = plt.subplots(2,1)
ax[0].plot(hist1.history['accuracy'], color='b', label="Training accuracy")
ax[0].plot(hist1.history['val_accuracy'], color='r',label="Validation accuracy")
legend = ax[0].legend(loc='best', shadow=True)

ax[1].plot(hist1.history['loss'], color='b', label="Training loss")
ax[1].plot(hist1.history['val_loss'], color='r', label="validation loss")
legend = ax[1].legend(loc='best', shadow=True)
```






Training recall
Validation recall
Training loss
validation loss


Training accuracy
Validation accuracy
Training loss
validation loss





A: SOURCE CODE

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# import seaborn as sns
import cv2
import os, glob
import tensorflow as tf
from tqdm import tqdm
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Model, Sequential, load_model
from tensorflow.keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D, BatchNormalization, AveragePooling2D,GlobalAveragePooling2D,Input
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import warnings
warnings.filterwarnings("ignore",category=DeprecationWarning)
```

```python
data_train = r"./Dataset/train"
data_test = r"./Dataset/validation"
fake=[]
for file in os.listdir(data_test):
    if file[0] =='.':
        continue
    fake+=[file]
print(fake)
print(len(fake))
# for file1 in os.listdir(data_train):
#     if file1[0] =='.':
#         continue
#     fake+=[file1]
# print(fake)
# print(len(fake))
```

```python
img_height,img_width=256,256
batch_size=20

train_datagen = ImageDataGenerator(rescale=1./255)
train_generator = train_datagen.flow_from_directory(data_train,
                                                    target_size=(img_height,img_width),
                                                    batch_size=batch_size,
                                                    class_mode='categorical',
                                                    )
test_generator = train_datagen.flow_from_directory(data_test,
                                                   target_size=(img_height,img_width),
                                                   batch_size=batch_size,
                                                   class_mode='categorical',

                                                   )
```

```python
precision=tf.keras.metrics.Precision()
recall=tf.keras.metrics.Recall()
sensitivity=tf.keras.metrics.SensitivityAtSpecificity(0.1)
specificity=tf.keras.metrics.SpecificityAtSensitivity(0.1)
```

```python
from sklearn.metrics import confusion_matrix
def plot_confusion_matrix(y_true, y_pred, classes,
                          normalize=False,
                          title=None,
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if not title:
        if normalize:
            title = 'Normalized confusion matrix'
        else:
            title = 'Confusion matrix, without normalization'

    # Compute confusion matrix
    cm = confusion_matrix(y_true, y_pred)
    # Only use the labels that appear in the data
    classes = classes
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)

    fig, ax = plt.subplots()
    im = ax.imshow(cm, interpolation='nearest', cmap=cmap)
    ax.figure.colorbar(im, ax=ax)
    # We want to show all ticks...
    ax.set(xticks=np.arange(cm.shape[1]),
```

```python
    fig, ax = plt.subplots()
    im = ax.imshow(cm, interpolation='nearest', cmap=cmap)
    ax.figure.colorbar(im, ax=ax)
    # We want to show all ticks...
    ax.set(xticks=np.arange(cm.shape[1]),
           yticks=np.arange(cm.shape[0]),
           # ... and label them with the respective list entries
           xticklabels=classes, yticklabels=classes,
           title=title,
           ylabel='True label',
           xlabel='Predicted label')

    # Rotate the tick labels and set their alignment.
    plt.setp(ax.get_xticklabels(), rotation=45, ha="right",
             rotation_mode="anchor")

    # Loop over data dimensions and create text annotations.
    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i in range(cm.shape[0]):
        for j in range(cm.shape[1]):
            ax.text(j, i, format(cm[i, j], fmt),
                    ha="center", va="center",
                    color="white" if cm[i, j] > thresh else "black")
    fig.tight_layout()
    return ax
```

```python
model = Sequential()

model.add(Conv2D(filters = 32, kernel_size = (5,5),padding = 'Same',
                 activation ='relu', input_shape = (img_height,img_width,3)))
model.add(Conv2D(filters = 32, kernel_size = (5,5),padding = 'Same',
                 activation ='relu'))
model.add(MaxPool2D(pool_size=(2,2)))


model.add(Conv2D(filters = 64, kernel_size = (3,3),padding = 'Same',
                 activation ='relu'))
model.add(Conv2D(filters = 64, kernel_size = (3,3),padding = 'Same',
                 activation ='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))

model.add(Flatten())
model.add(Dense(256, activation = "relu"))

model.add(Dense(6, activation = "softmax"))

model.summary()
```