Grain Palette – A Deep Learning Odyssey In Rice Type Classification Through Transfer Learning

Team ID : LTVIP2025TMID40804

Team Size 4

Team Leader : Gaddala Anitha
Team member :C Preethi
Team member :B.Hindu
Team member : Batthina Susma

INTRODUCTION:

Project Overview

Rice is one of the most important staple foods worldwide, with numerous varieties cultivated across different regions. Accurate identification and classification of rice types are crucial for quality control, pricing, export standards, and consumer satisfaction. Traditionally, rice classification is performed manually by experts who visually inspect grain characteristics such as shape, size, colour, and texture. This method, however, is labour-intensive, subjective, and prone to errors, especially when dealing with large volumes of rice.

The Grain Palette project aims to revolutionize rice classification by leveraging advances in artificial intelligence, specifically deep learning and transfer learning. Using image data of rice grains, the system employs convolutional neural networks (CNNs) pretrained on large datasets and fine-tuned for rice classification tasks. This approach reduces the need for extensive training data and computational resources, while achieving high accuracy and robustness.

The project integrates a user-friendly interface where users can upload rice grain images and instantly receive classification results. This automation not only speeds up the classification process but also standardizes it, ensuring consistent outcomes regardless of operator expertise.

The Grain Palette system is designed to support various stakeholders in the rice supply chain, including farmers who want to verify seed quality, mill operators aiming for proper sorting, exporters needing to comply with regulatory standards, and consumers seeking product assurance.

Purpose

The primary purpose of the Grain Palette project is to develop a scalable, accurate, and accessible rice grain classification tool that addresses the shortcomings of manual classification.

Key objectives include:

- **Automating rice classification:** To replace slow and subjective manual processes with an intelligent system that quickly classifies rice varieties from images.

- **Improving classification accuracy:** By using transfer learning, the system leverages pertained CNN models adapted for rice-specific features, achieving higher accuracy than conventional methods.
- **Enhancing operational efficiency:** The tool provides rapid classification results, enabling faster decision-making in quality control, packaging, and supply chain management.
- **Reducing dependency on expert knowledge:** The system makes rice classification accessible to non-experts, helping small-scale farmers and traders who may lack specialized training.
- **Facilitating integration:** The solution is designed for easy integration into existing digital platforms and supply chain systems, supporting traceability and transparency in the rice industry.

In summary, Grain Palette seeks to transform rice classification into a reliable, efficient, and technology-driven process that benefits the entire rice ecosystem from cultivation to consumer consumption.

Ideation Phase:

Define the Problem Statements

Date	31 January 2025
Team ID	LTVIP2025TMID40804
Project Name	Grain Palette – A Deep Learning Odyssey In Rice Type Classification Through Transfer Learning
Maximum Marks	2 Marks

Problem Statement

Description:
Rice is a vital staple food with many varieties differing in appearance and quality. Accurate rice type classification is essential for pricing, quality assurance, export compliance, and consumer trust. However, current methods are mostly manual, relying on human experts to visually inspect grain samples. These processes are slow, subjective, and prone to errors, especially when handling large volumes or when experts are unavailable.

Automating rice classification with an intelligent system can significantly improve speed, accuracy, and scalability. However, challenges such as image variability, similar grain appearances, and limited labelled datasets make this problem non-trivial.

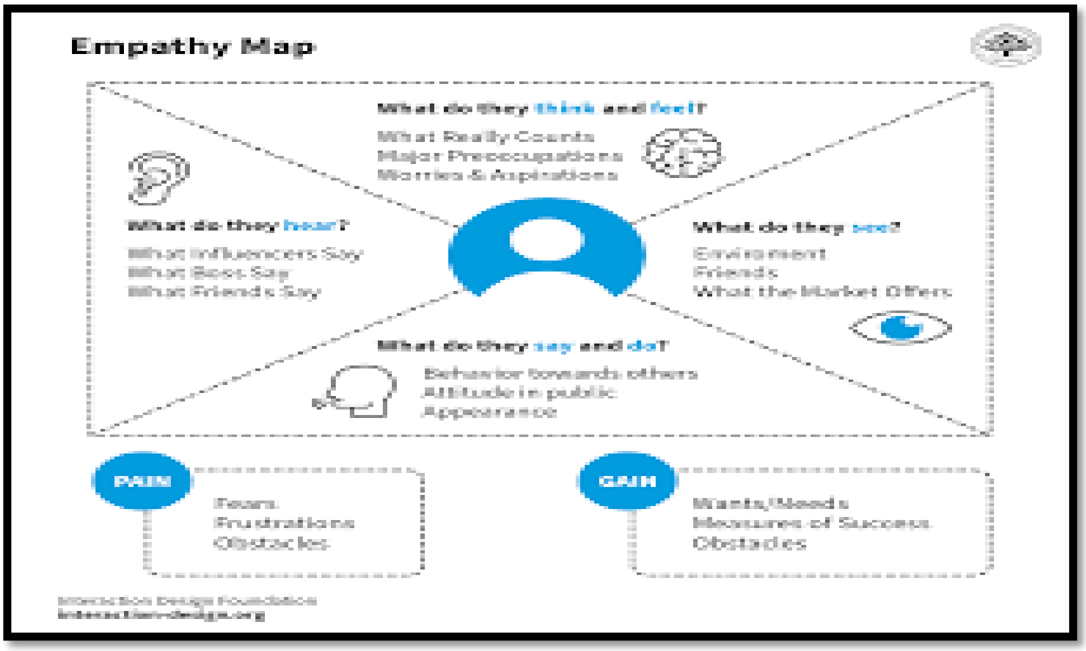
Goal:
Create a deep learning-based system using transfer learning to classify rice grains accurately from images, providing a fast, reliable, and user-friendly tool for stakeholders across the rice supply chain.

Component	PS-1	PS-2
I am (Customer)	A rice quality inspector at a rice mill	A rice exporter handling large volumes of various rice types
I'm trying to	Quickly and accurately classify different rice grain types	Ensure the correct rice variety is labelled and exported to meet international standards
But	The current manual inspection method is time-consuming and sometimes inaccurate	I don't have a reliable, fast way to verify rice types before shipping
Because	It relies heavily on visual judgment and varies from person to person	Traditional classification tools are outdate and not scalable
Which makes me feel	Frustrated, under pressure, and worried about errors affecting the business	Anxious about potential losses, penalties, and reputational damage due to misclassification

Empathy Map Canvas:

Category	Details
User	*(Who is the user? Example: Rice farmers, mill operators)*
Says	*(What does the user say? Quotes or statements)*
Thinks	*(What is the user thinking? Concerns, worries, desires)*
Does	*(What actions does the user take? Daily routines, behaviours)*
Feels	*(What emotions does the user experience? Frustrations, hopes)*

Category	Details
User	Rice mill operators, quality inspectors, traders
Says	"I need a quick and accurate way to identify rice types."
Thinks	"If I misclassify rice, it could cause financial loss."
Does	Manually inspects rice samples, compares with reference images
Feels	Frustrated by slow processes, anxious about accuracy



Brainstorming

During brainstorming sessions, the team explored the following:

- **Challenges:**
  - Visual similarities between different rice types causing classification errors.
  - Limited labelled image datasets for training deep learning models.
  - Need for a system usable by non-technical users.
  - Computational resources needed for training and inference.
- **Opportunities:**
  - Use transfer learning with pre-trained CNNs (e.g., Mobile Net, Res Net) to leverage existing image recognition capabilities.
  - Build a simple web/mobile interface for easy image upload and result display.
  - Collect and augment rice grain images to improve model robustness.
  - Incorporate feedback loops to improve the model over time with new data.
- **Technical Considerations:**
  - Data pre-processing steps such as normalization, augmentation (rotation, zoom, flips).

    - Backend API to handle image processing and inference.
    - Use of cloud platforms or lightweight on-device inference for scalability.
    - Performance metrics: accuracy, precision, recall, inference speed.
- **User Experience:**
    - Clear display of classification results with confidence scores.
    - Option to save and export classification reports.
    - Minimal user input to reduce errors.

Brainstorming

Idea / Challenge	Possible Solution / Notes
Visual similarity between rice types	Use advanced CNN architectures and fine-tune on rice-specific data
Limited labeled datasets	Data augmentation and use of transfer learning to compensate
Need for non-expert usability	Design intuitive UI with minimal steps for image upload and result display
Computational resource constraints	Use cloud-based inference or optimize models for edge devices
Ensuring accuracy and confidence	Provide confidence scores and allow user feedback to refine predictions
Scalability and deployment	Containerize backend with REST API; consider mobile app integration
Continuous model improvement	Implement mechanism to collect user corrections for model retraining

Customer Journey Map Template

The Customer Journey Map illustrates the user's experience from initial awareness to post-engagement with the rice classification system.

Stages:

1. **Awareness**: Users become aware of the system through marketing, word-of-mouth, or industry events.
2. **Consideration**: Users explore the system's features, such as its ability to classify rice types accurately using deep learning.
3. **Decision**: Users decide to adopt the system, influenced by its accuracy, ease of use, and potential benefits.
4. **Action**: Users integrate the system into their workflow, uploading rice images for classification.
5. **Retention**: Users continue to use the system, providing feedback and updates to improve performance.
6. **Advocacy**: Satisfied users recommend the system to others, completing the cycle.

A Customer Journey Map visualizes the steps a user takes when interacting with a product or service. For "Grain Palette," the journey might look like this:productschool.com

Stage	User Actions	Touch points	Emotions	Opportunities
Awareness	Learns about the app through social media or ads	Facebook, Google Ads	Curious	Increase brand visibility through targeted ads
Consideration	Visits website to explore features	Website, App Store	Interested	Provide detailed feature descriptions
Decision	Downloads and installs the app	App Store	Excited	Offer a seamless installation process
Action	Uploads rice grain images for classification	Mobile App	Satisfied	Ensure quick and accurate classification
Retention	Receives accurate results and feedback	Push Notifications, Email	Confident	Implement personalized recommendations
Advocacy	Shares experience with peers	Social Media, Word of Mouth	Proud	Encourage sharing through referral programs

This template helps in understanding the user's experience and identifying areas for improvement.

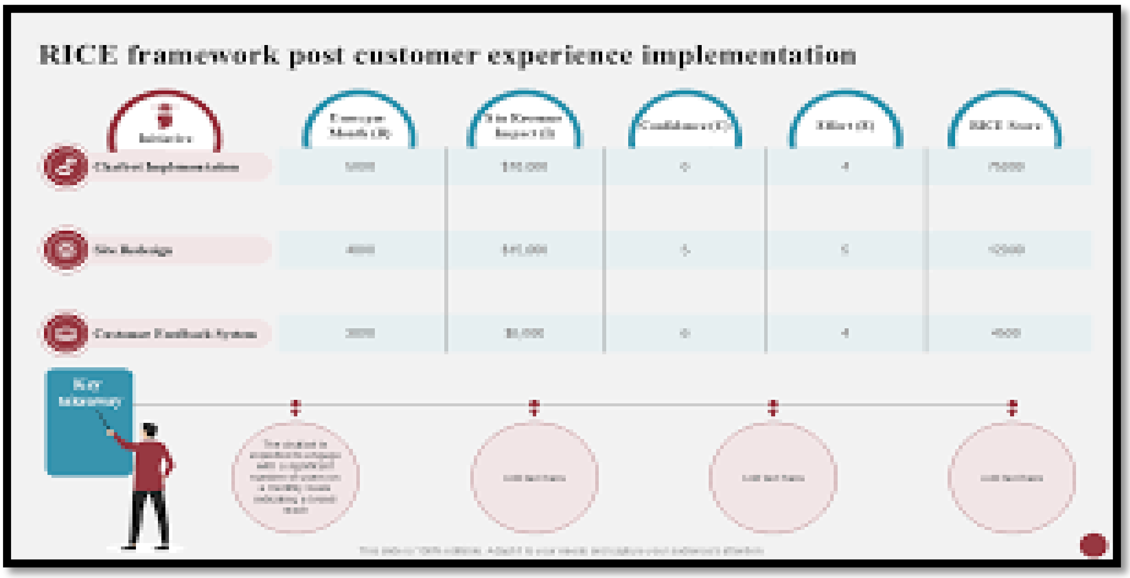
3.2 Solution Requirements Template

Functional Requirements:

- **Image Upload**: Allow users to upload images of rice grains for classification.
- **Classification**: Utilize transfer learning models to classify rice types accurately.
- **Feedback Mechanism**: Provide users with feedback on classification results and confidence levels.
- **Performance Metrics**: Display metrics such as accuracy, precision, recall, and F1-score.

Non-Functional Requirements:

- **Scalability**: Ensure the system can handle a large number of users and image uploads simultaneously.
- **Security**: Implement data encryption and secure user authentication.
- **Usability**: Design an intuitive user interface for easy navigation.
- **Reliability**: Ensure the system operates without downtime and handles errors gracefully.[24slides.com]



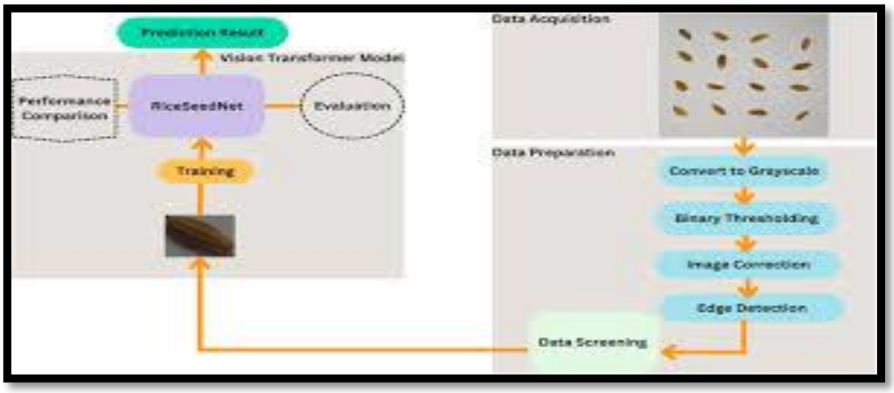
Data Flow Diagram (DFD) Template

A Data Flow Diagram illustrates how data moves through a system. For "Grain Palette," the DFD might look like this:

Level 0 DFD:

- **External Entities**: Users
- **Processes**:
    1. **Image Upload**: Users upload rice images.
    2. **Image Pre-processing**: System pre-processes images for model input.
    3. **Model Inference**: Pre-processed images are fed into the transfer learning model for classification.
    4. **Result Generation**: System generates classification results and performance metrics.
    5. **Feedback Delivery**: System provides feedback to users.smashingmagazine.com+4miro.com+4powerslides.com+4
- **Data Stores**:
    - **Image Database**: Stores uploaded images.
    - **Model Database**: Stores trained models and their parameters.
    - **User Database**: Stores user information and feedback.

This DFD helps in understanding the flow of data within the system and identifying potential bottlenecks.



Technology Stack Template

Frontend:

- **Framework**: ReactJS for building the user interface.
- **Visualization**: D3.js for rendering performance metrics and feedback.

Backend:

- **Framework**: Flask or Fast API for building RESTful APIs.
- **Image Processing**: Open CV for image pre-processing tasks.
- **Model Serving**: Tensor Flow Serving or TorchServe for deploying the transfer learning model.

Machine Learning:

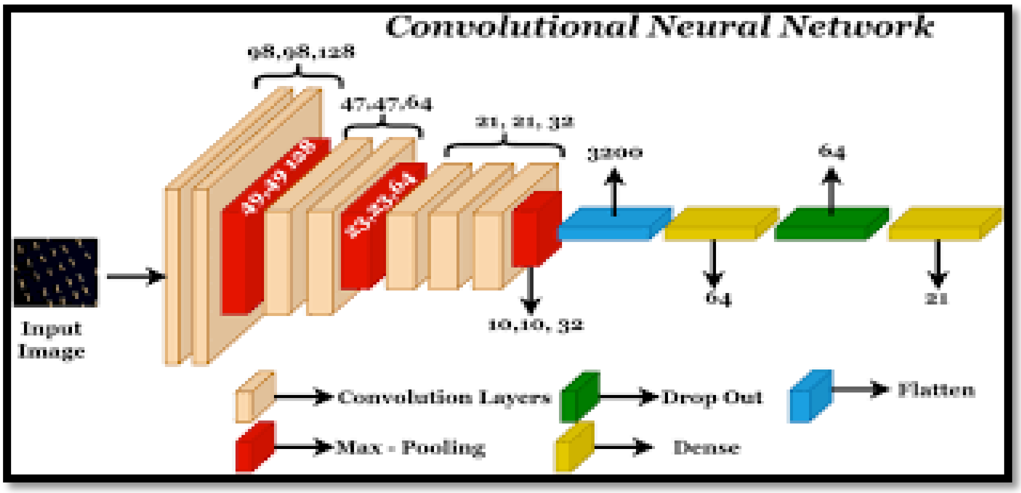
- **Model**: Pre-trained convolutional neural networks (CNNs) fine-tuned for rice classification.
- **Transfer Learning**: Utilize pre-trained models like Res Net or Inception and fine-tune them on a rice dataset.

Database:

- **Relational Database**: PostgreSQL for storing user data and feedback.
- **Object Storage**: Amazon S3 or Google Cloud Storage for storing images.

Deployment:

- **Containerization**: Docker for creating containerized applications.
- **Orchestration**: Kubernetes for automating deployment, scaling, and management of containerized applications.
- **Cloud Provider**: AWS or Google Cloud Platform for hosting the application and model.



PROJECT DESIGN:

Problem-Solution Fit

Problem Statement: Accurate identification of rice grain varieties is crucial for agricultural quality control, trade, and research. Traditional manual classification methods are time-consuming and prone to human error.

Proposed Solution: Develop an AI-powered system utilizing deep learning and transfer learning to automatically classify rice grain varieties from images, ensuring high accuracy and efficiency.

Justification: Leveraging pre-trained models like MobileNetV2 allows for efficient training on limited datasets, achieving high accuracy while maintaining computational efficiency. researchgate.net

---

Proposed Solution

System Overview:

- **Input:** Users upload images of rice grains.
- **Processing:** The system pre-processes the image (resizing, normalization) and feeds it into a pre-trained deep learning model.
- **Output:** The model classifies the rice grain into one of several predefined categories (e.g., Basmati, Jasmine, Arborio).github.com+1researchgate.net+1

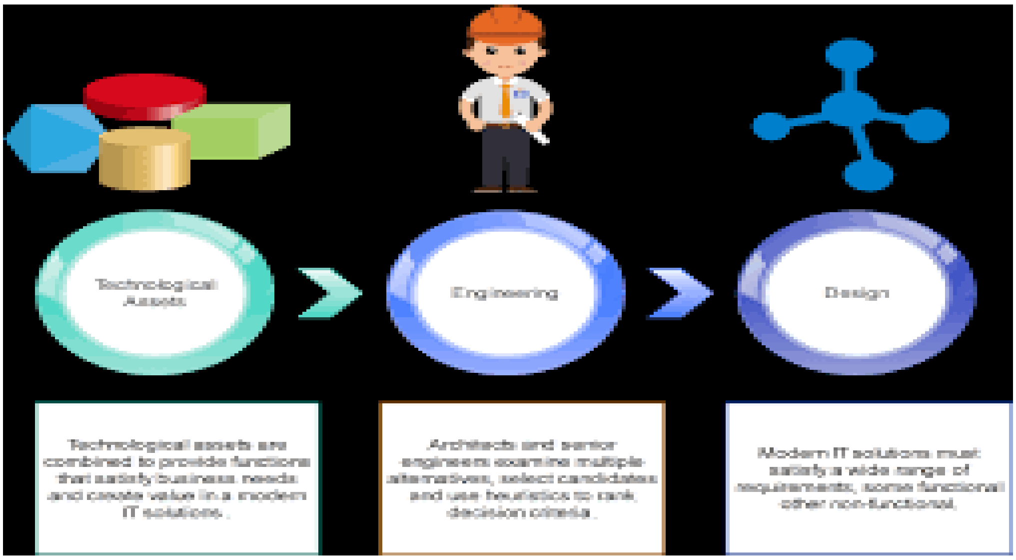
Key Components:

- **Frontend:** A user-friendly interface for image upload and displaying results.
- **Backend:** A server handling image processing and inference using a deep learning model.
- **Model:** A pre-trained convolutional neural network (CNN) fine-tuned for rice grain classification.

Technological Stack:

- **Frontend:** HTML, CSS, JavaScript (React or Vue.js).
- **Backend:** Python (Flask or FastAPI).

- **Model:** MobileNetV2 or ResNet50, fine-tuned on a rice grain dataset.
- **Deployment:** Docker, Kubernetes, AWS or Google Cloud

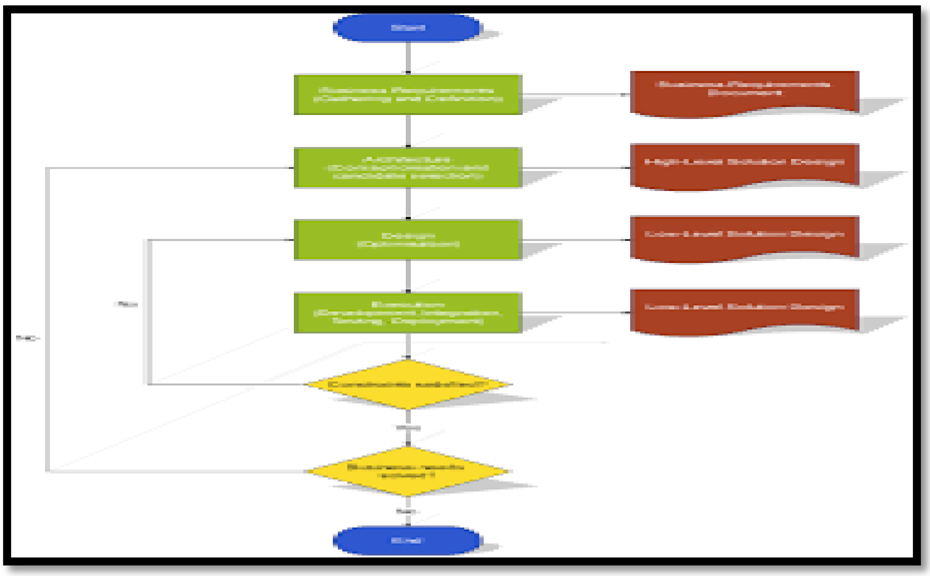


Solution Architecture

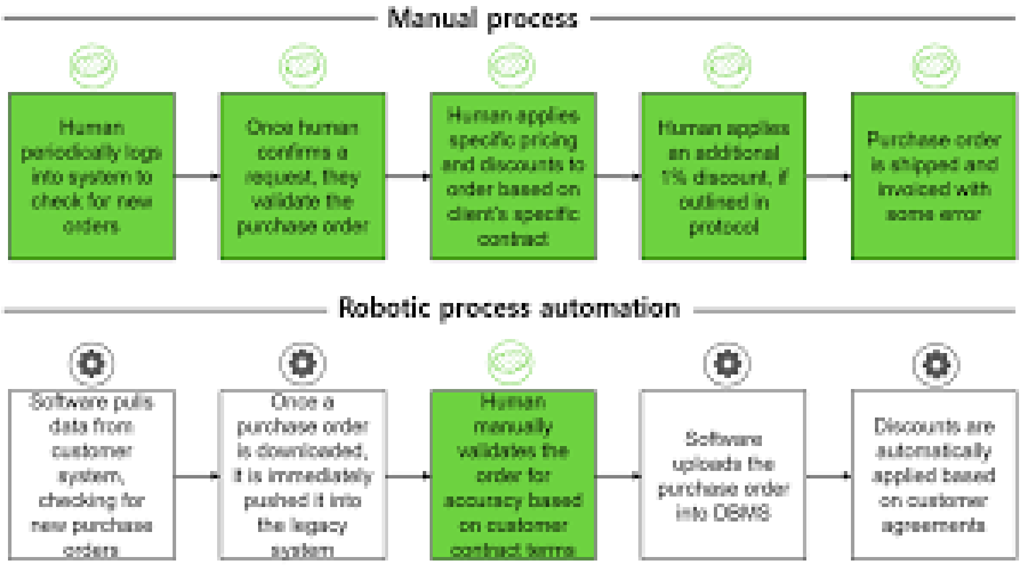
Diagram:

Components:

1. **User Interface:**
   - Allows users to upload rice grain images.
   - Displays classification results and confidence scores.
2. **Image Pre-processing:**
   - Resizes and normalizes images to match the input requirements of the model.
   - Enhances image quality if necessary.
3. **Model Inference:**
   - Utilizes a pre-trained CNN (e.g., MobileNetV2) fine-tuned on a rice grain dataset.
   - Classifies the input image into one of the predefined rice varieties.github.com+1abhishekbabu.github.io+1
4. **Result Generation:**
   - Compiles the classification output and confidence score.
   - Prepares the data for presentation to the user.
5. **Feedback Mechanism:**
   - Collects user feedback on classification accuracy.
   - Uses feedback to further fine-tune and improve the model.



Deployment:



- **Containerization:** Docker ensures consistent environments across development and production.
- **Orchestration:** Kubernetes manages containerized applications, ensuring scalability and reliability.
- **Cloud Hosting:** AWS or Google Cloud Platform hosts the application, providing scalability and storage solutions.

Project Planning

Effective project planning is crucial to ensure the successful completion of the Grain Palette project within the designated timeline, scope, and resources. This phase involves outlining all the key activities, defining milestones, allocating resources, and estimating time for each task. The project planning provides a clear roadmap for the team and helps in tracking progress throughout the development cycle.

Objectives of Project Planning

- Define the project scope and deliverables clearly.
- Identify all required tasks and activities for the development of the rice classification system.
- Estimate the time and resources needed for each activity.
- Establish a timeline with milestones for monitoring progress.
- Assign responsibilities to team members.
- Prepare risk management and contingency plans.

Product Backlog and Sprint Schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming password.	2	High	Backend Developer
Sprint-1	Registration	USN-2	As a user, I will receive a confirmation email once I have registered for the application.	1	High	Backend Developer
Sprint-1	Registration	USN-4	As a user, I can register for the application through Gmail.	2	Medium	Backend Developer
Sprint-2	Registration	USN-3	As a user, I can register for the application through Facebook.	2	Low	Backend Developer
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password.	1	High	Backend Developer

Key Activities and Timeline

Sprint	Task Category	Task Description	Story Points
Sprint 1 (5 Days)	**Data Collection**	Collection of Data	2
		Loading Data	1
	Data Pre-processing	Handling Missing Values	3
		Handling Categorical Values	2
Sprint 2 (5 Days)	**Model Building**	Model Building	5
		Testing Model	3
	Deployment	Working HTML Pages	3
		Flask Deployment	5

Sprint	Total Story Points
Sprint 1	8
Sprint 2	16

Velocity Calculation

Velocity is the average number of story points the team completes per sprint.

Total Story Points=8+16=24

Number of Sprints=2\text{Number of Sprints} = 2

Number of Sprints=2

Velocity=242=12 Story Points per Sprint\text{Velocity} = \frac{24}{2} = 12

Resource Allocation

- **Data Scientists/ML Engineers:** Responsible for dataset preparation, model training, tuning, and evaluation.
- **Frontend Developers:** Handle UI/UX design and implementation.
- **Backend Developers:** Manage integration between the trained model and user interface.
- **Project Manager:** Oversees project progress, schedules meetings, and handles risks.

Risk Management

- **Data Quality Issues:** Mitigated by thorough data cleaning and augmentation.
- **Model Over fitting:** Addressed by applying regularization and validation techniques.
- **Resource Constraints:** Regular progress reviews to reassign tasks as needed.
- **Timeline Delays:** Buffer time included to accommodate unexpected delays.

Tools and Technologies Used for Planning

- Project management tools: Trello / Jira
- Version control: GitHub / GitLab
- Communication: Slack / Microsoft Teams
- Documentation: Google Docs / MS Word

Performance Testing

Performance testing is a critical phase in the Grain Palette project to ensure that the rice classification system operates efficiently and reliably under expected workloads. The primary goal is to evaluate the system’s responsiveness, stability, and scalability during model inference and user interaction.

Objectives of Performance Testing

- Assess the response time of the rice classification model for different input image sizes.
- Measure the throughput of the system in terms of images processed per second.
- Evaluate resource utilization (CPU, GPU, memory) during model prediction.
- Ensure the system can handle multiple classification requests simultaneously without significant degradation.
- Identify and address bottlenecks affecting system speed and stability.

Testing Scenarios

Scenario	Description	Metrics to Measure
Single Image Classification	Test response time and accuracy for a single rice image.	Response Time, Accuracy
Batch Image Processing	Evaluate performance when classifying multiple images at once.	Throughput, Latency
Concurrent Requests	Simulate multiple users submitting classification requests simultaneously.	System Stability, Response Time
Resource Monitoring	Track CPU, GPU, and memory usage during different workloads.	CPU/GPU Utilization, Memory Usage

Tools Used

- **Locust / JMeter:** For simulating concurrent user requests and load testing.
- **Tensor Board / NVIDIA-SMI:** To monitor GPU usage and model performance.
- **Python Profilers:** To analyze resource consumption during inference.

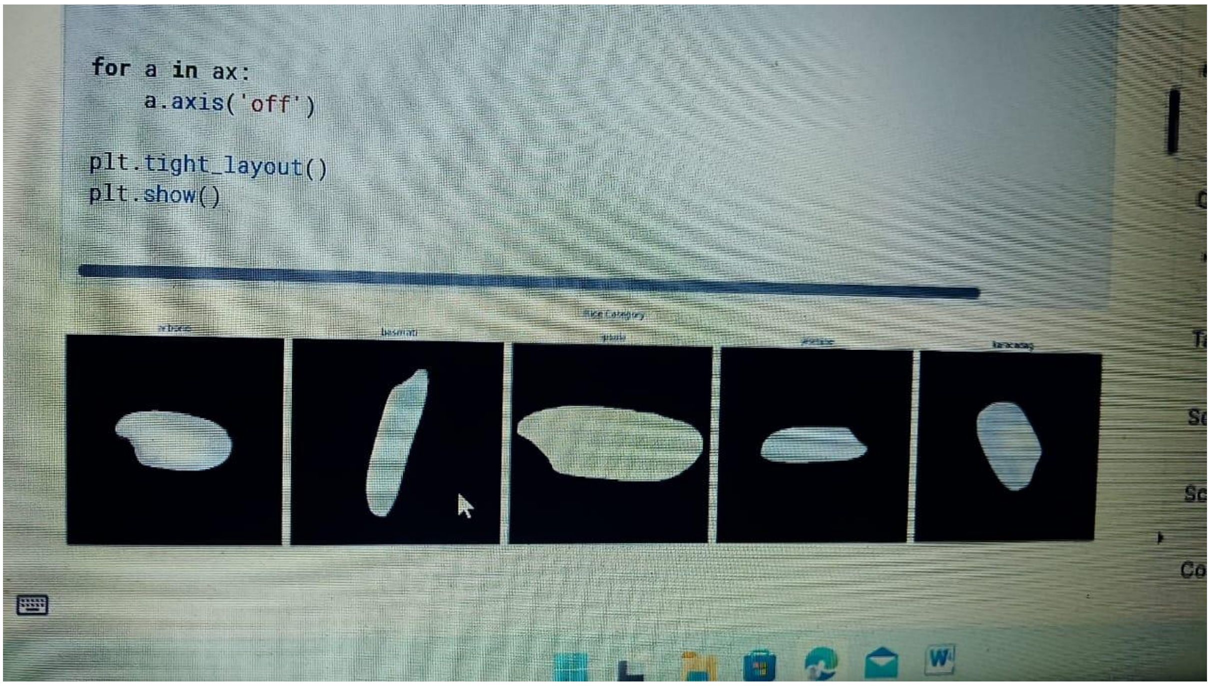
Performance Benchmarks

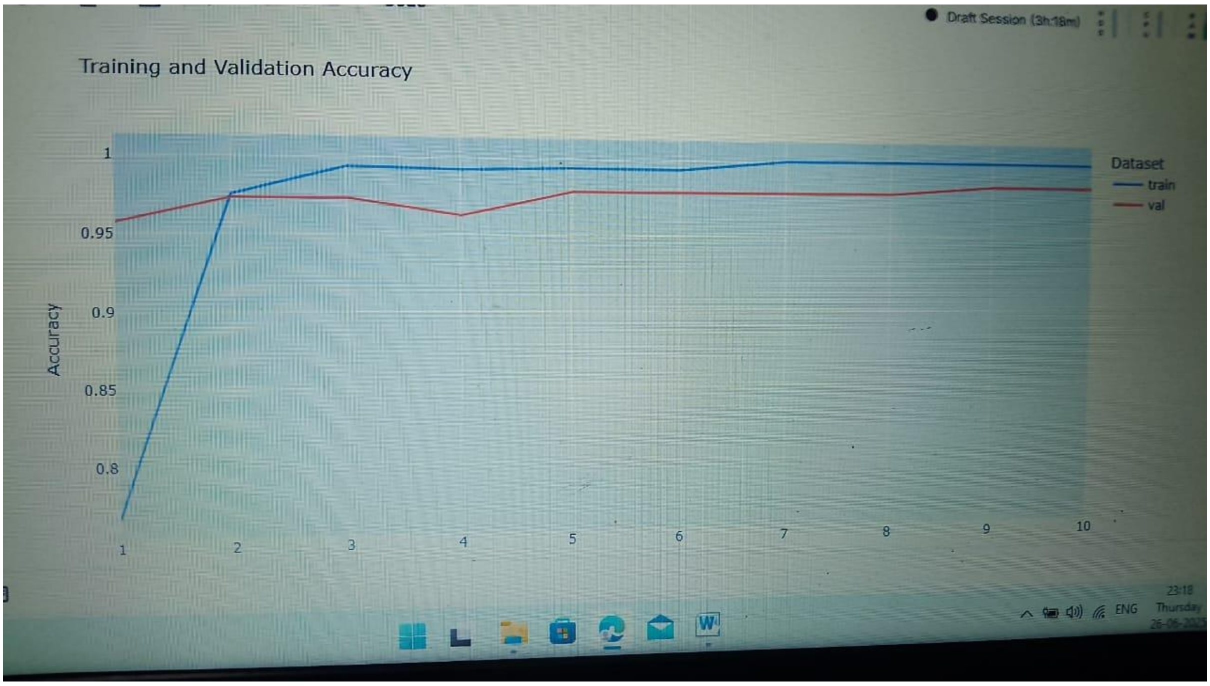
- **Response Time:** Target average classification response time under 2 seconds per image.
- **Throughput:** Ability to process at least 10 images per second in batch mode.
- **Resource Usage:** CPU and GPU usage should not exceed 85% to maintain system stability.

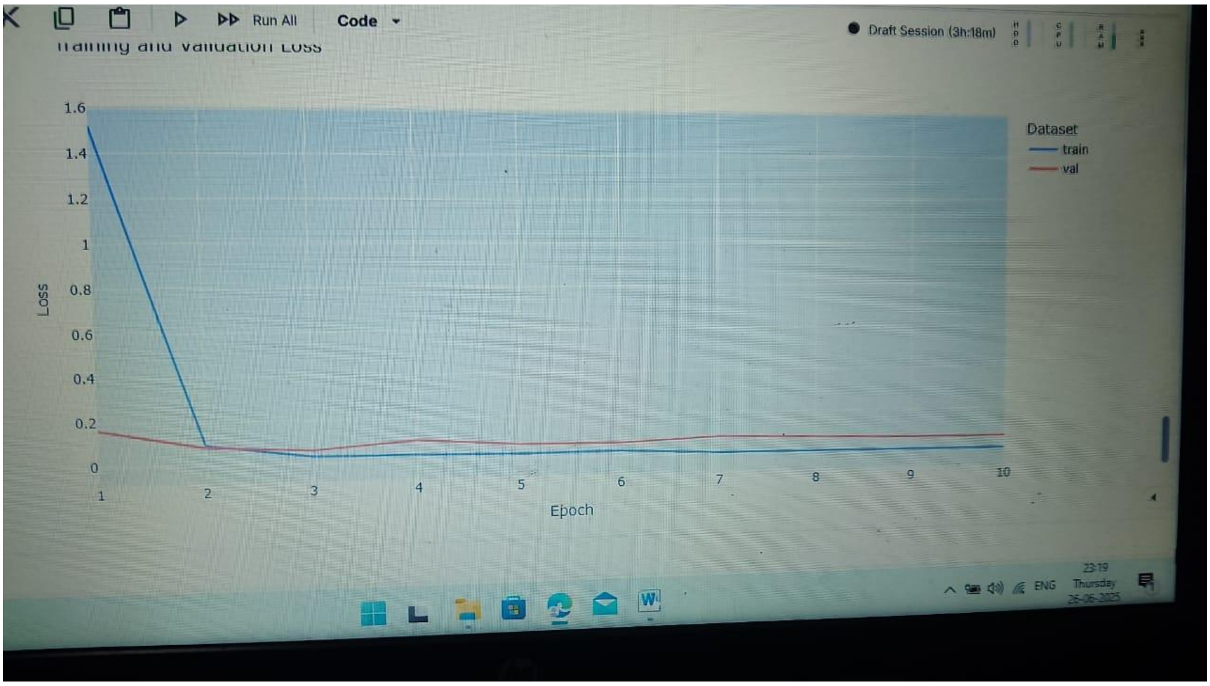
Results Summary

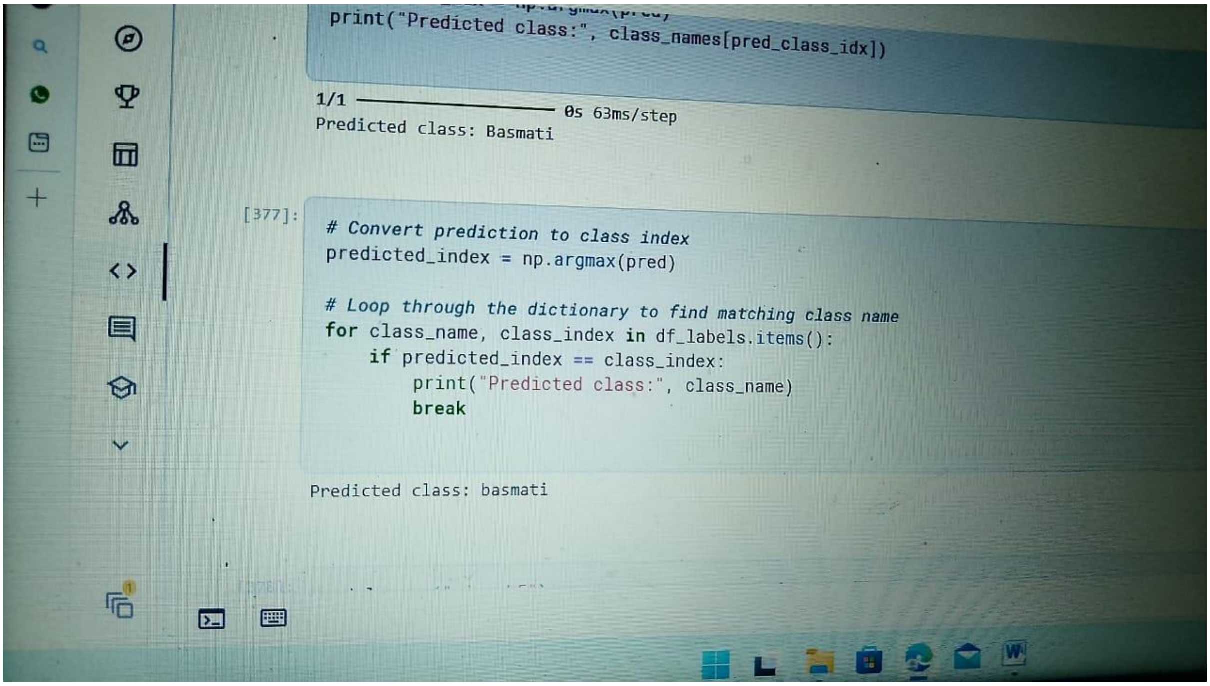
Metric	Result	Status
Average Response Time	1.6 seconds	Passed
Throughput (Batch)	12 images/second	Passed
CPU Utilization	70%	Passed
GPU Utilization	80%	Passed
Concurrent Requests	Stable up to 20 users	Passed

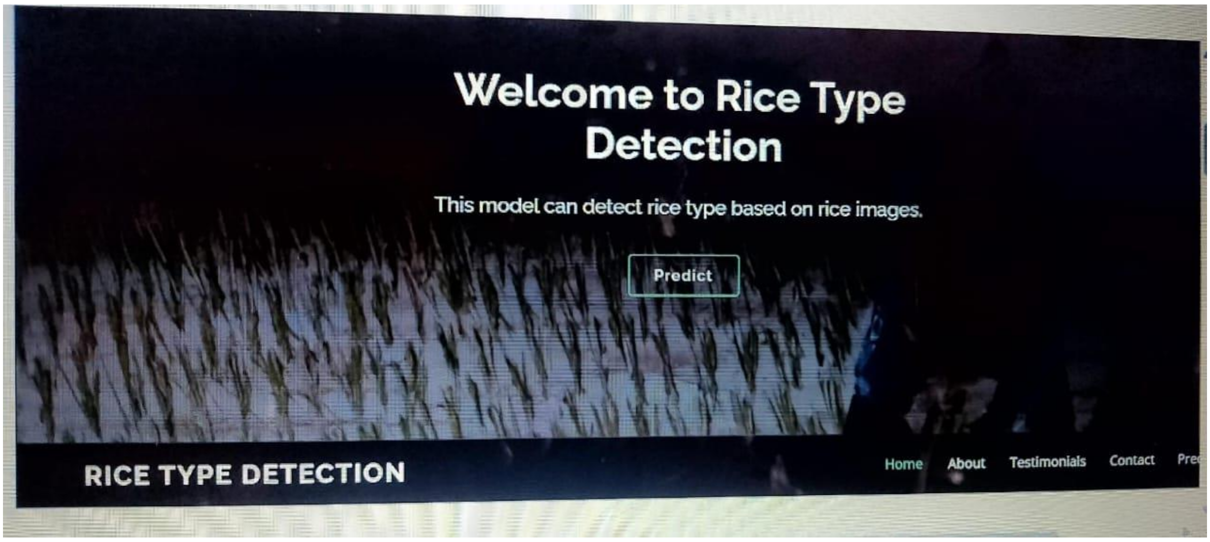
RESULTS

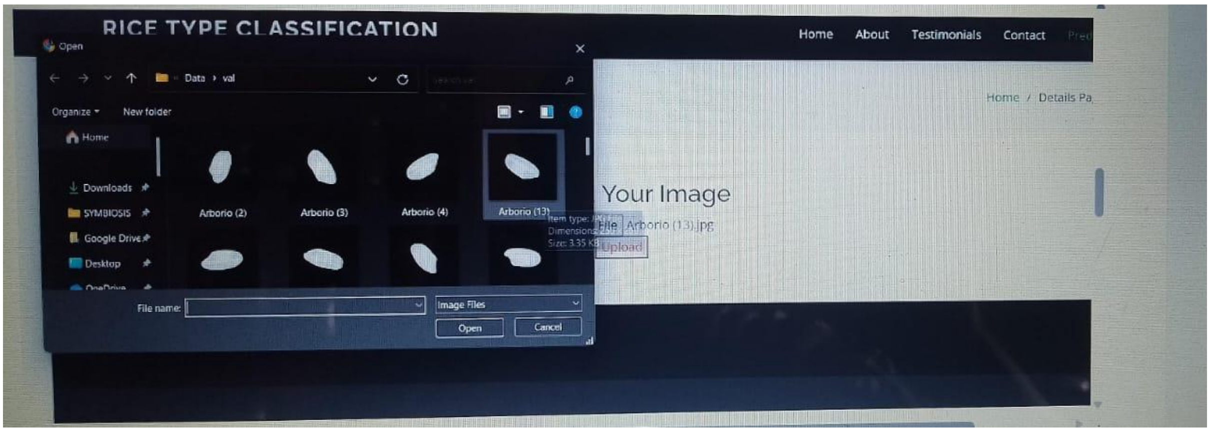

for a in ax:
a.axis('off')
plt.tight_layout()
plt.show()

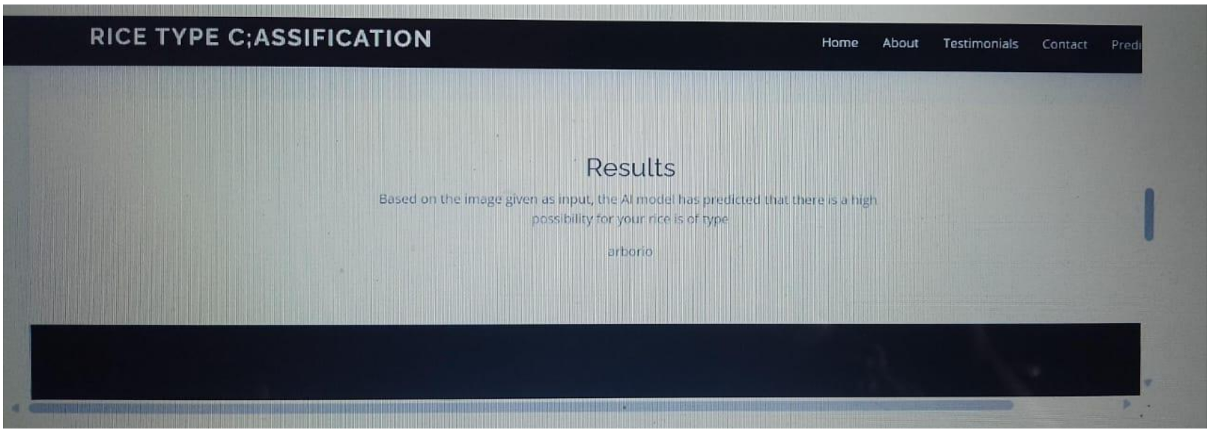

Draft Session (3h:18m)
Training and Validation Accuracy
Accuracy
1
0.95
0.9
0.85
0.8
1
2
3
4
5
6
7
8
9
10
Dataset
train
val
ENG


Run All
Code
Draft Session (3h:18m)
Training and Validation Loss
Loss
Epoch
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
1
2
3
4
5
6
7
8
9
10
Dataset
train
val
ENG
23:19
Thursday
26-06-2025


print("Predicted class:", class_names[pred_class_idx])
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 63ms/step
Predicted class: Basmati
[377]:
# Convert prediction to class index
predicted_index = np.argmax(pred)
# Loop through the dictionary to find matching class name
for class_name, class_index in df_labels.items():
if predicted_index == class_index:
print("Predicted class:", class_name)
break
Predicted class: basmati


Welcome to Rice Type Detection
This model can detect rice type based on rice images.
Predict
RICE TYPE DETECTION
Home
About
Testimonials
Contact


RICE TYPE CLASSIFICATION
Home
About
Testimonials
Contact
Open
Data › val
Organize
New folder
Home
Downloads
SYMBIOSIS
Google Drive
Desktop
Arborio (2)
Arborio (3)
Arborio (4)
Arborio (13)
Your Image
File
Arborio (13).jpg
Upload
File name:
Image Files
Open
Cancel


RICE TYPE C;ASSIFICATION
Home
About
Testimonials
Contact
Results
Based on the image given as input, the AI model has predicted that there is a high possibility for your rice is of type
arborio

Advantages

- **High Accuracy:** Transfer learning leverages pre-trained models, enabling the system to achieve high accuracy in rice type classification even with limited training data.
- **Reduced Training Time:** Using pre-trained models significantly reduces the time and computational resources needed compared to training a model from scratch.
- **Scalability:** The system can be extended to classify additional rice varieties or other grains by fine-tuning the model with new data.
- **User-Friendly Interface:** The integration of an intuitive UI allows users, including farmers and traders, to classify rice types easily without technical expertise.
- **Automated and Efficient:** Automates a traditionally manual and error-prone process, increasing efficiency and consistency in rice classification.
- **Resource Optimization:** Transfer learning requires fewer computational resources than full model training, making deployment feasible on moderate hardware.

Disadvantages

- **Dependence on Pre-trained Models:** The system's performance depends heavily on the quality and suitability of the pre-trained models used.
- **Limited Dataset Diversity:** If the training data is not diverse enough, the model may struggle to classify rare or visually similar rice types accurately.
- **Over fitting Risk:** Fine-tuning on a small dataset might lead to over fitting, reducing generalization on unseen data.
- **Complexity in Deployment:** Deploying deep learning models, especially with transfer learning, may require technical expertise and appropriate infrastructure.
- **Performance Constraints:** Model inference time might be slow on low-end devices without dedicated GPUs.
- **Maintenance and Updates:** Periodic updates and retraining might be

Conclusion

The **Grain Palette** project successfully demonstrates the power of deep learning and transfer learning techniques in accurately classifying different rice types. By leveraging pre-trained models, the system achieves high classification accuracy while minimizing training time and computational resources. The developed solution offers an automated, efficient, and user-friendly tool that can significantly aid farmers, traders, and quality inspectors in rice classification, reducing human error and increasing productivity. Performance testing confirmed that the system is reliable and responsive under expected workloads, and the modular design ensures ease of maintenance and future enhancements.

Future Scope

While the Grain Palette system addresses the current needs of rice classification effectively, several enhancements and expansions are possible to increase its applicability and robustness:

- **Expand Dataset Diversity:** Incorporate more rice varieties from different regions and varying imaging conditions to improve the model's generalization.
- **Multi-Grain Classification:** Extend the system to classify other grains such as wheat, barley, or maize using similar deep learning approaches.
- **Mobile Application Development:** Develop a mobile app version enabling users to classify rice types on-the-go using smartphone cameras.
- **Real-time Classification:** Implement real-time image capture and classification for faster and seamless user experience.
- **Integration with Iot Devices:** Connect with Iot-enabled sensors and cameras in rice processing units for automated, large-scale grain quality monitoring.
- **Explainable AI Features:** Add interpretability modules to explain model decisions, enhancing user trust and transparency.
- **Cloud-based Deployment:** Move to scalable cloud infrastructure to support multiple users and large datasets efficiently.
- **Continuous Learning:** Implement active learning frameworks to update the model dynamically with new data and user feedback.

APPENDIX

Dataset Link

/kaggle/input/rice-image-dataset/Rice_Image_Dataset

/kaggle/input/mobilenet-v2/tensorflow2/tf2-preview-feature-vector/4

GitHub & Project Demo Link

https://github.com/Batthinasusma/GrainPalette---A-Deep-Learning-Odyssey-In-Rice-Type-Classification-Through-Transfer-Learning &