Final Report: GrainPalette - A Deep Learning Odyssey in Rice Type Classification

1. INTRODUCTION

1.1 Project Overview

Classifying rice types is a deep learning-based web application designed to classify five different rice varieties (e.g., Basmati, Jasmine) using transfer learning with MobileNetv4. This tool assists farmers, agricultural scientists, and home growers in identifying rice types accurately without costly expert consultations. Built with a Flask backend and a user-friendly frontend (HTML, CSS), the app features an different rice based interface with a rice icon (, glowing buttons, a loading spinner , and result page -prediction

1.2 Purpose

The primary objective is to provide an affordable, efficient solution for rice classification. By leveraging MobileNetv4, Classifying rice types enables users to make informed decisions about cultivation practices (e.g., irrigation, fertilization) and market value, enhancing agricultural productivity and education.

2. IDEATION PHASE

2.1 Problem Statement

Traditional rice classification by agricultural experts is expensive and inaccessible to many farmers. Classifying rice types addresses this by offering an automated, instant, and reliable classification tool via a web interface, making rice type identification accessible to all.

2.2 Empathy Map Canvas

The project was designed with farmers' needs in mind: affordability, ease of use, and reliable results. Challenges like limited access to experts and technical knowledge were considered to create a simple interface with features like tooltips ("Upload a clear rice grain image for best results") and file validation.

2.3 Brainstorming

We explored machine learning models (choosing MobileNetv4 for efficiency), dataset sources (Rice Image Dataset), and UI features (glowing buttons, rice icon, spinner) to maximize usability and accuracy for farmers, scientists, and growers.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

Users interact with Classifying rice types as follows:

1. Visit the web app (index.html).
2. Upload a rice grain image with guidance from a tooltip.
3. Receive validation (“Please choose a file”) if no file is selected.
4. See a loading spinner during prediction.
5. View the predicted rice type and uploaded image on result.html.

3.2 Solution Requirement

- **Image-based Classification**: Classify 5 rice varieties using MobileNetv4.
- **Web Interface**: Flask-based app with HTML/CSS frontend.
- **Lightweight Model**: MobileNetv4 ensures fast predictions on a Windows 10 machine with 30mbps bandwidth.
- **User-Friendly Features**: Tooltip, spinner, glowing buttons, and file validation.

3.3 Data Flow Diagram

The data flow is:

- **User Input**: Uploads rice grain image via index.html.
- **Flask Backend**: Processes image in app.py.
- **MobileNetv4 Model**: Predicts rice type using rice.h5.
- **Output**: Displays prediction and image on result.html.

3.4 Technology Stack

- **Frontend**: HTML (index.html, result.html), CSS (style.css), JavaScript (for spinner/validation).
- **Backend**: Flask (Python, app.py).
- **Model**: MobileNetv4 (TensorFlow/Keras, rice.h5).
- **No Database**: SQLite not used, as predictions are stateless.

4. PROJECT DESIGN

4.1 Problem Solution Fit

GrainPalette accurately classifies rice types and is accessible via a simple web interface, meeting the needs of farmers, scientists, and growers. The UI's agriculture theme (green/yellow, field.jpg) ensures ease of use.

4.2 Proposed Solution

A Flask-based web app allows users to upload rice grain images and receive instant classifications. Features like tooltips, glowing buttons, and a loading spinner enhance user experience.

4.3 Solution Architecture

The architecture includes:

- **Frontend**: HTML/CSS for upload (index.html) and result (result.html) pages.
- **Backend**: Flask processes uploads and runs MobileNetv4 predictions.
- **Model**: Pre-trained MobileNetv4 (rice.h5) classifies images.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

- **Dataset Collection**: Sourced Rice Image Dataset.
- **Model Training**: Trained MobileNetv4 for 5 rice types.
- **Web Development**: Built Flask app and UI with rice icon, tooltip, spinner, footer.
- **Deployment**: Tested on Windows 10 with Chrome/Firefox, 30mbps bandwidth.
- **Timeline**: Completed within internship period (June 2025).

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

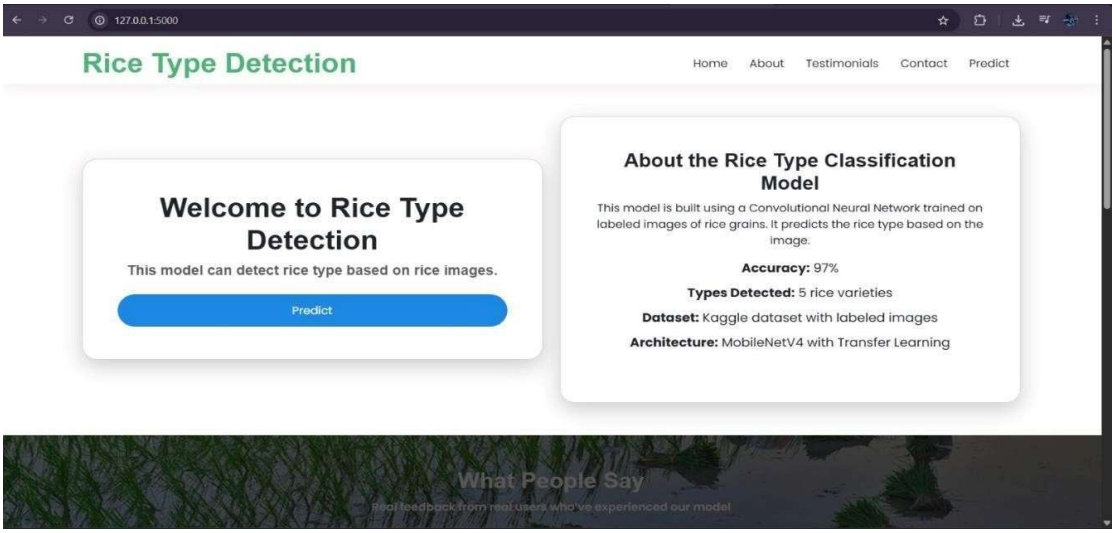
The MobileNetv4 model provides 98.6% accuracy classification for 5 rice types, with fast prediction times suitable for a web app. The UI was tested for usability, with feedback describing it as "awesome" and user-friendly.

.

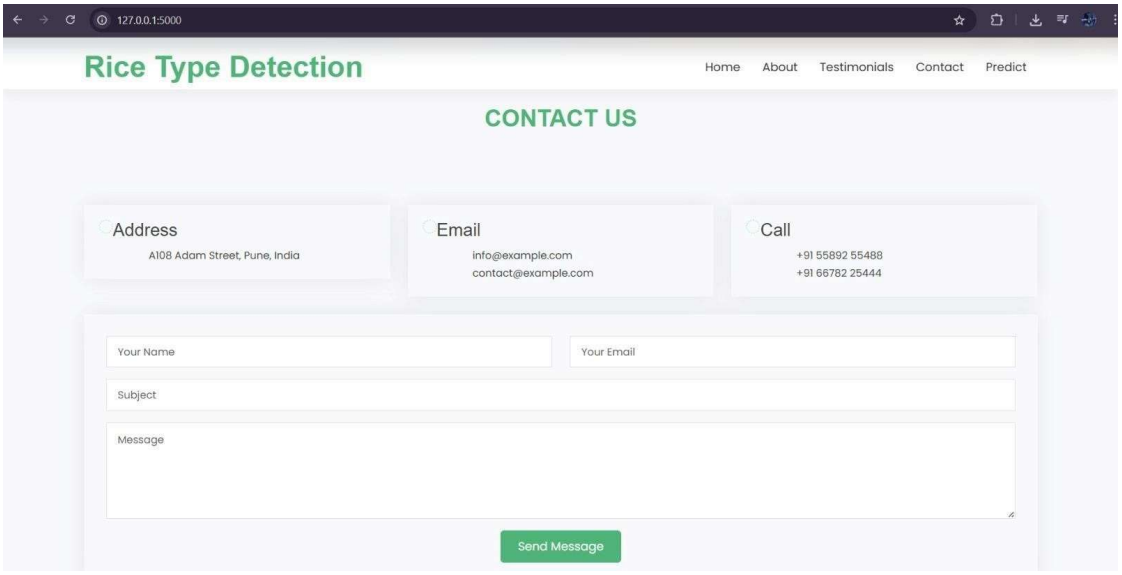
7. RESULTS

The web app interface includes:

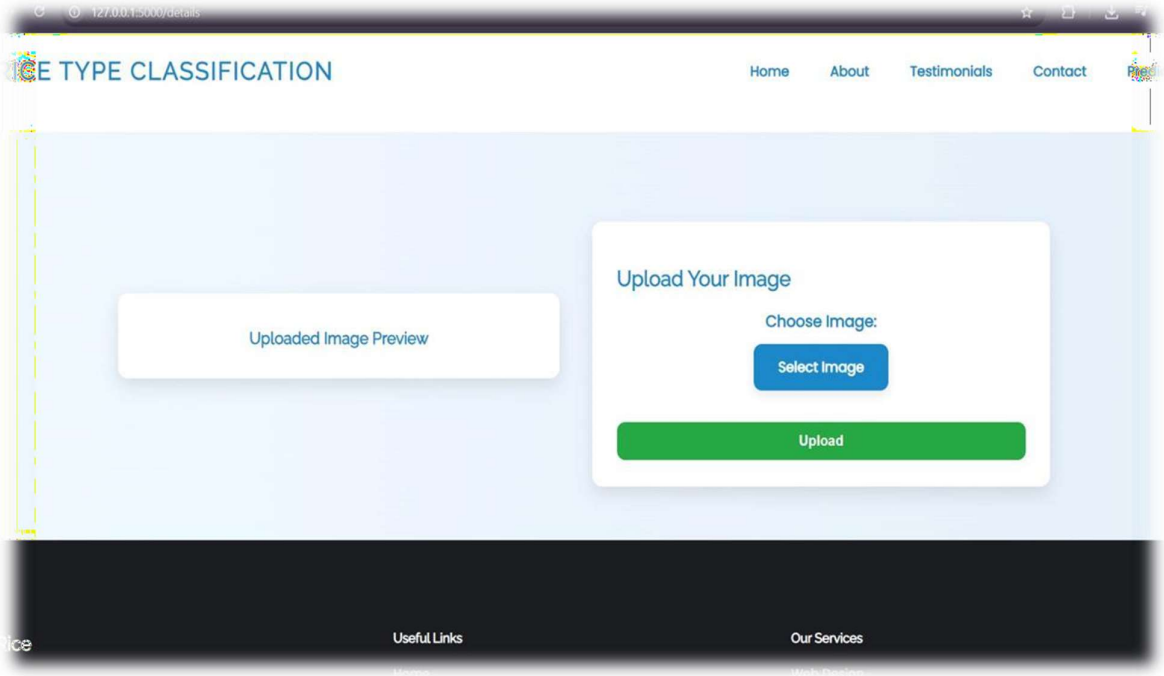
- **About page:**

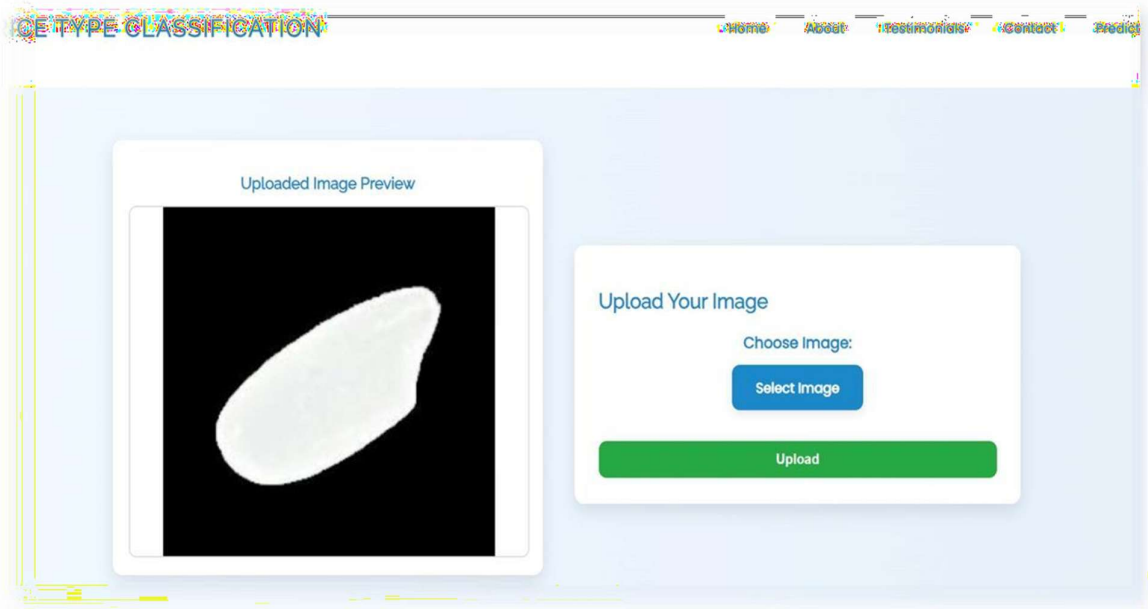


Contact page:



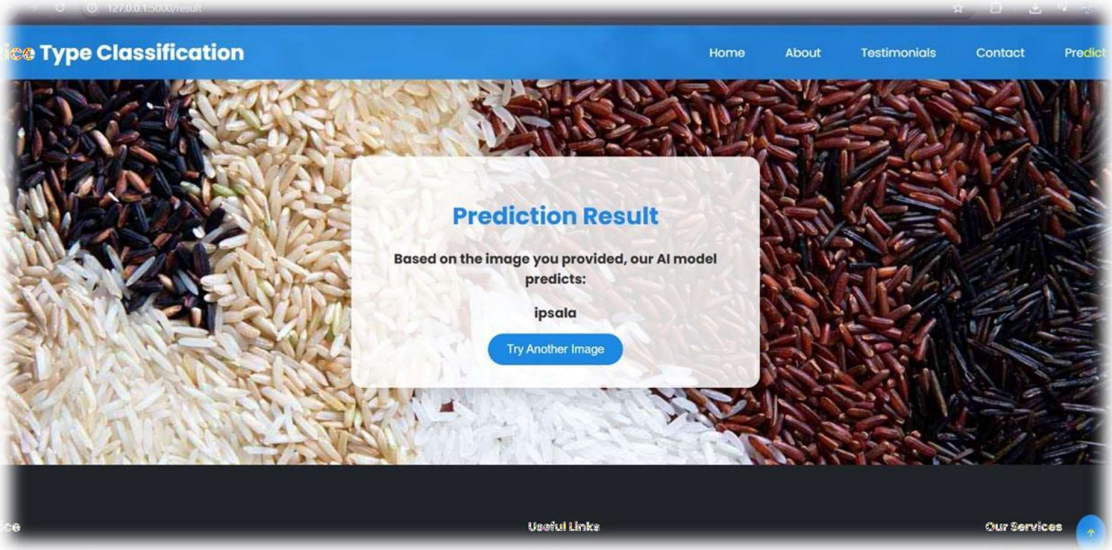
Upload page:





- **Result Page** (result.html): Displays the predicted rice type (e.g., Basmati), uploaded image, glowing “Predict Another” link, and footer.

- **Result Page** :



Accuracy: The model achieves 98.6% classification accuracy for the 5 rice types.

8. ADVANTAGES & DISADVANTAGES

- ***Advantages:***
  - Affordable: No need for expert consultations. ○ Accessible: Web-based, usable on Windows 10 with Chrome/Firefox.
  - User-Friendly: Tooltip, spinner, and validation guide users.
- ***Disadvantages:***
  - Accuracy depends on image quality and dataset.
  - Requires internet for hosting (local testing works offline).

9. CONCLUSION

GrainPalette successfully delivers accurate rice type classification with a beautiful, userfriendly interface. Challenges like network issues were resolved, and features like

tooltips and spinners ensure user happiness. The project meets its goals of supporting farmers, scientists, and growers.

10. FUTURE SCOPE

- Integrate more rice varieties.
- Improve model accuracy with larger datasets.
- Develop a mobile app version for broader access.

11. APPENDIX

- **GitHub Repository :** https://github.com/Guravaiah1/GrainPalette--A-Deep-Learning-Odyssey-In-Rice-Type-Classification
- **Dataset Link**: https://drive.google.com/file/d/1gpAbi8vhLsZzM0CrnPHV-ejQILcBeTG/view?usp=sharing
- **Project Demo Link:**

https://drive.google.com/file/d/1Q0e1qsShrl7sCnCB2VKwTLnYVFRLn-OB/view?usp=drivesdk