Artificial Intelligence & Machine Learning Virtual Internship Program

Conducted by

Smart Bridge

PROJECT REPORT

Submitted by:

V. Venu
V. Pujitha
Y. Revanth
Y. Krishnam Raju

Grain Palette - Rice Type Classification Using Deep Learning

Abstract

Rice is one of the most widely consumed grains globally. Its quality and market value often depend on the genetic variety, which can be identified by characteristics like color, size, and shape. This project, titled 'GrainPalette', uses Convolutional Neural Networks (CNN) and Transfer Learning (MobileNetV2) to classify five rice varieties: Arborio, Basmati, Ipsala, Jasmine, and Karacadag. The project includes both model training and deployment as a user-friendly web app using Flask.

Introduction

With a growing population, demand for rice is constantly increasing. Ensuring quality and variety classification is key for market value and customer satisfaction. Manual inspection is time-consuming and subjective. Therefore, an AI-based system that can classify rice grains using image data offers a faster and scalable solution.

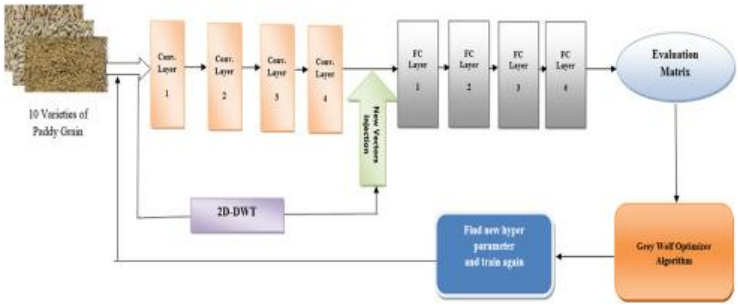
Techniques Used

Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs) are a class of deep learning models specifically designed for analyzing visual data like images. They work by automatically learning spatial features (such as edges, shapes, and textures) from input images through a series of layers.

In a CNN:

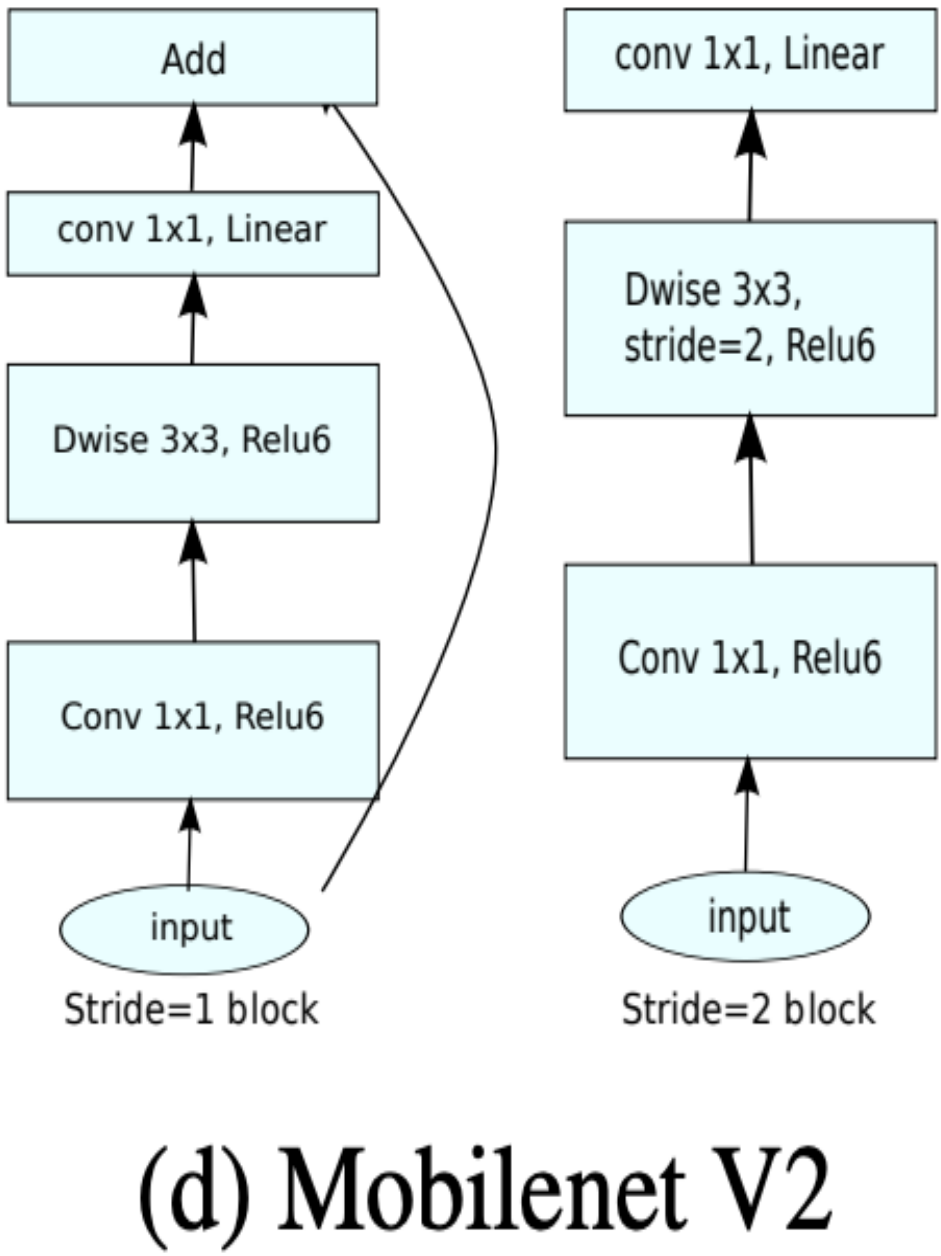
- **Convolutional Layers** apply filters (or kernels) to input images to extract low-level features (e.g., lines, curves).
- **Activation Functions** (like ReLU) introduce non-linearity so the model can learn complex patterns.
- **Pooling Layers** (usually Max Pooling) reduce the dimensionality of feature maps, making the model faster and more generalizable.
- **Fully Connected Layers** near the end combine all extracted features to make the final classification.



MobileNetV2

MobileNetV2 is a lightweight and efficient deep learning architecture developed by Google, designed specifically for mobile and embedded vision applications. It is an improved version of the original Mobile Net, known for delivering high accuracy while being computationally less expensive.

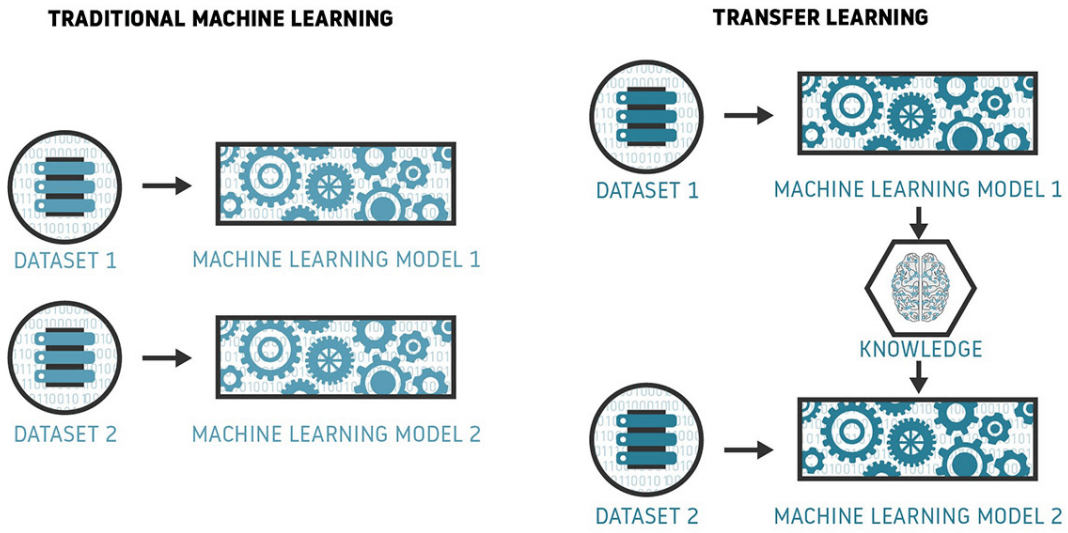
MobileNetV2 uses a technique called depth wise separable convolutions, which breaks down the standard convolution operation into two simpler steps—depth wise and pointwise convolutions. This drastically reduces the number of parameters and computation required.



(d) Mobilenet V2

Transfer Learning

Transfer learning is a deep learning approach where a model developed for one task is reused as the starting point for a model on a different but related task. In this project, we use MobileNetV2—pre-trained on the ImageNet dataset—as a feature extractor to classify rice grains. Instead of training a neural network from scratch, we retain the powerful visual features learned by MobileNetV2 and add a custom classification layer on top for identifying five rice varieties. This reduces the need for a large dataset, shortens training time, and improves accuracy, making it ideal for agricultural applications with limited labeled data.



TensorFlow

TensorFlow is an open-source deep learning framework developed by Google, widely used for building, training, and deploying machine learning models. In this project, TensorFlow serves as the core tool for constructing the rice classification model using layers from MobileNetV2 and custom dense layers for final prediction. It provides powerful APIs to handle data preprocessing, model architecture definition, compilation, training, and saving the trained model as rice.h5. TensorFlow also ensures compatibility with both CPUs and GPUs, enabling faster training and easy integration with web applications through saved models.

Flask

Flask is a lightweight and flexible web framework written in Python, commonly used for building web applications

and APIs. In this rice classification project, Flask is used to create a simple yet effective web interface that allows users to upload rice grain images for prediction. The uploaded image is processed and passed to the pre-trained deep learning model, and the predicted rice variety is displayed back to the user along with the image. Flask handles routing between pages, manages form submissions, and integrates seamlessly with HTML templates, making it ideal for deploying machine learning models in a user-friendly and interactive way.

HTML/CSS

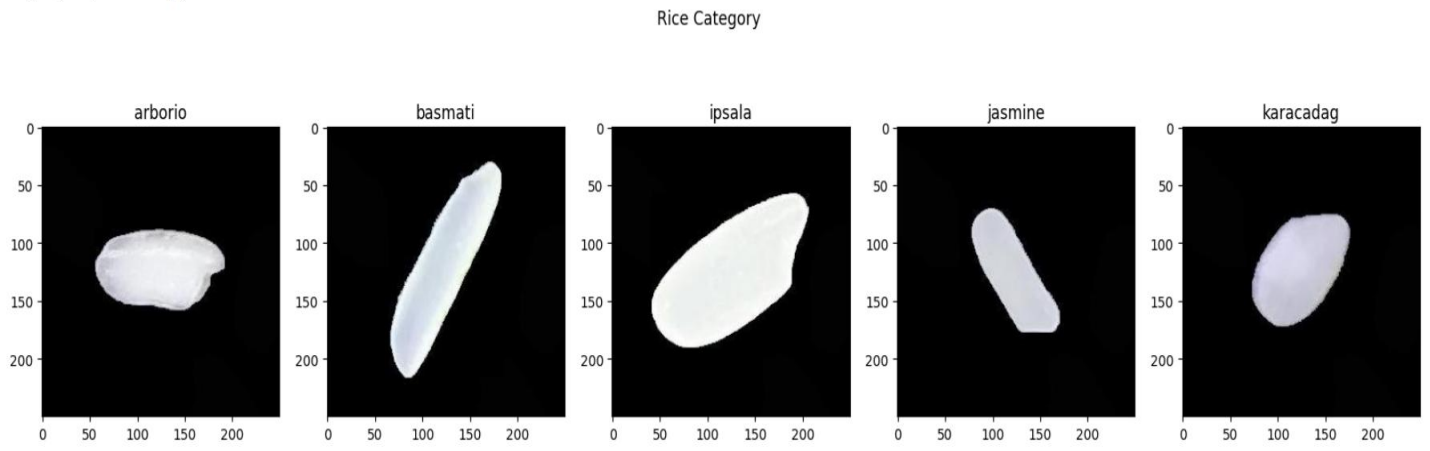
HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets) form the foundation of the frontend interface in this project. HTML structures the content of the web pages—such as image upload forms and result displays—while CSS enhances the user experience by adding styling, layout, and responsiveness. Together, they create a clean and user-friendly interface where users can interact with the rice classification model through a web browser.

Matplotlib & Plot

Matplotlib and Polly are Python libraries used for data visualization. In this project, Matplotlib is used to generate static plots of training and validation accuracy/loss, while portly provides interactive graphs for a more engaging display. These visualizations help evaluate the model's performance over epochs and make it easier to analyse trends, detect overfitting, or validate improvements, which is crucial for understanding model behaviour during training.

Dataset Description

The dataset contains 55 images from five rice types, each having 150 samples. These include morphological and color-based features. Images were resized to 224x224 and normalized for model compatibility.



Model Architecture

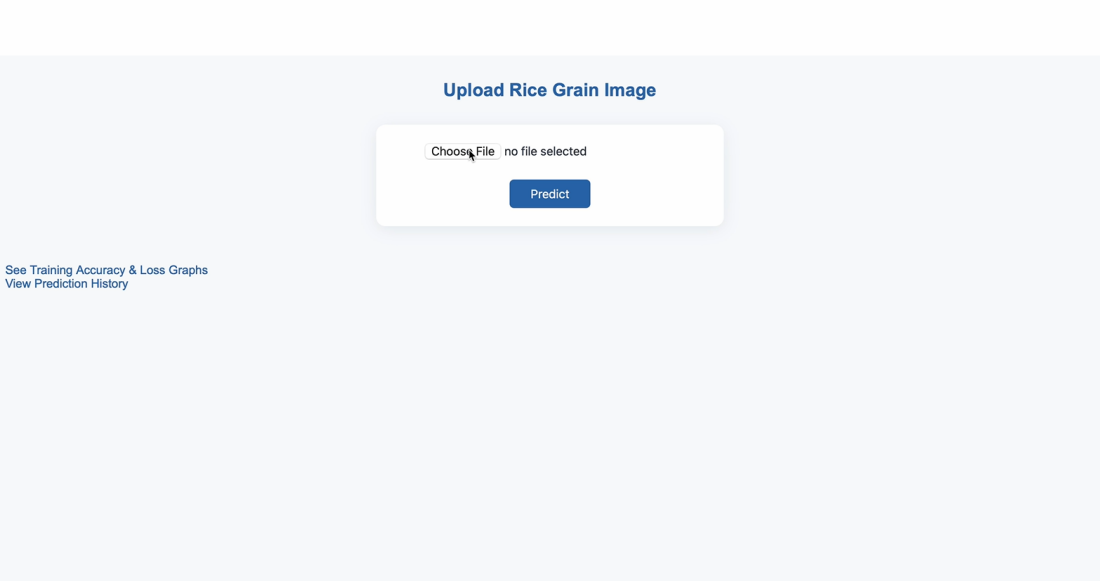
MobileNetV2 is used as the backbone for transfer learning. It is followed by:
- GlobalAveragePooling2D
- Dropout Layer

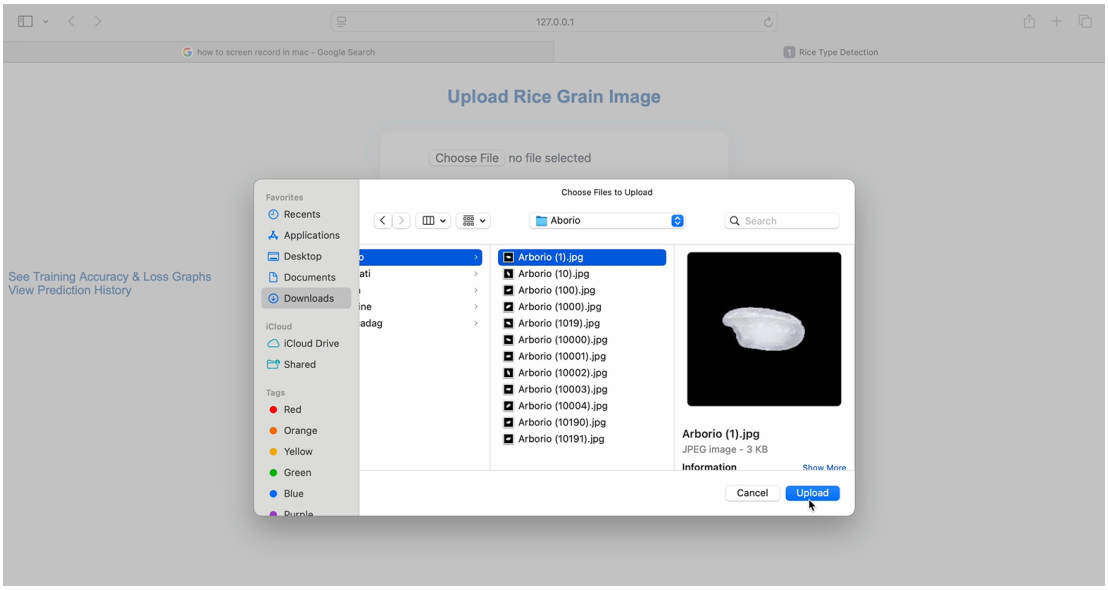
- Dense Layer with 5 softmax outputs

The model is trained for 10 epochs with Adam optimizer and Sparse Categorical Cross entropy loss.

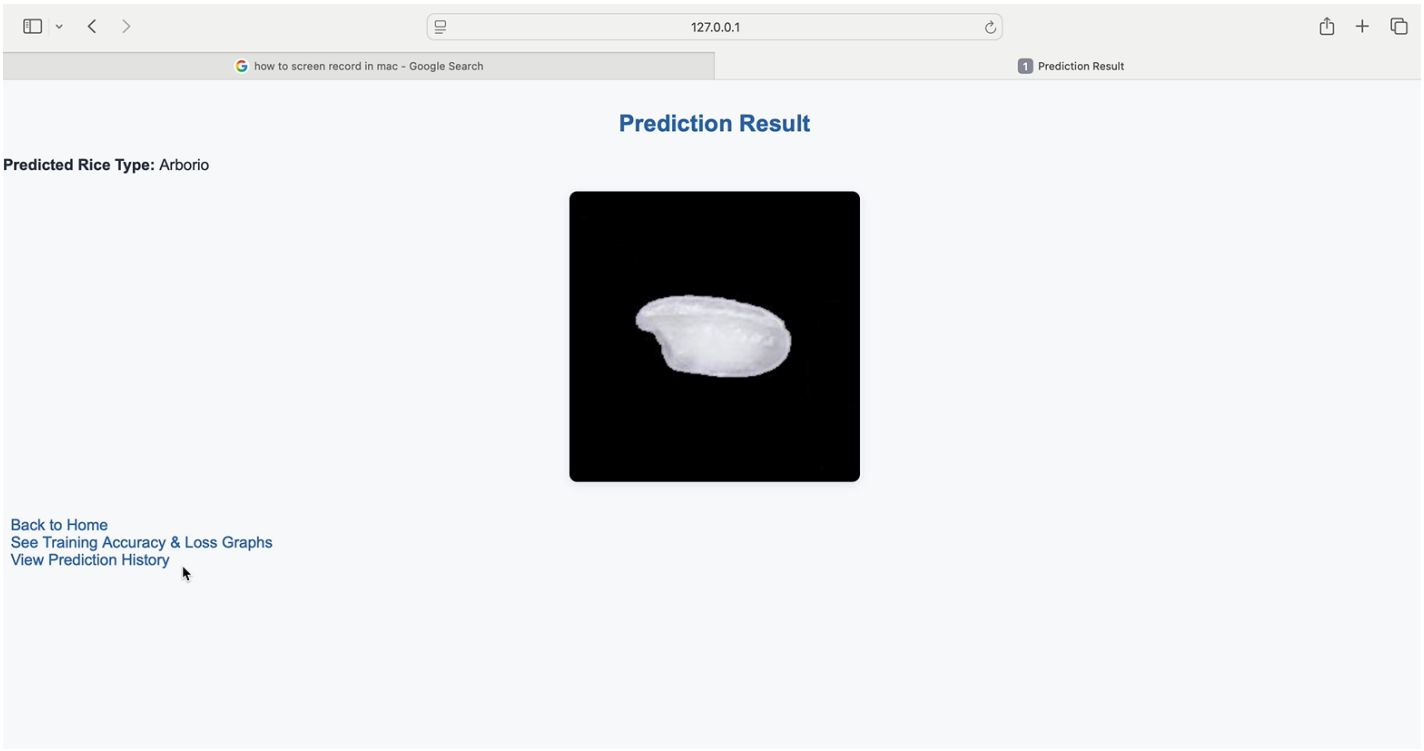
Web Application:

The Grain Palette web application is developed using Flask for backend processing and HTML/CSS for the frontend interface. It allows users to upload rice grain images, which are then resized, normalized, and passed to the trained MobileNetV2 model for prediction. The predicted rice variety is displayed along with the uploaded image on a result page, providing instant feedback. Flask also maintains a session-based prediction history and includes additional pages to visualize training accuracy and loss. This lightweight and user-friendly design ensures easy local use and supports future deployment on platforms like Heroku or AWS, making the system accessible and practical for real-world agricultural applications.





Index page and Details page

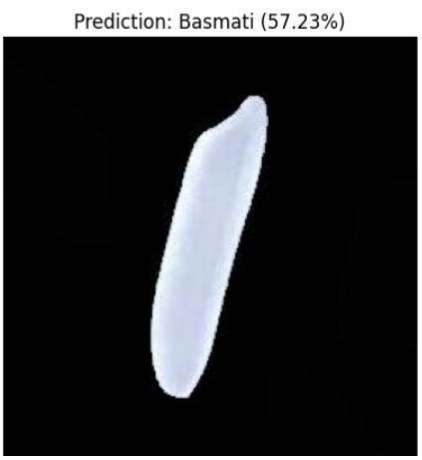


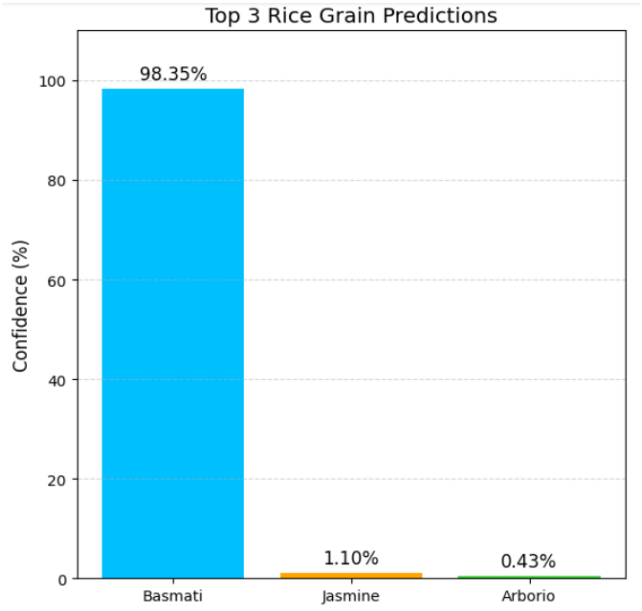
Results Page

Results and Evaluation:

The rice grain classification model built using MobileNetV2 achieved approximately 60% accuracy on the validation dataset. Although the performance is lower than expected, it serves as a solid foundation for further improvements. Training and validation accuracy and loss curves were plotted to monitor model behavior over epochs. The plots indicate that the model is learning but may require more data, better preprocessing, or fine-tuning to achieve higher accuracy. Despite its limitations, the model is functional and integrated into a working web application, demonstrating the feasibility of deep learning for agricultural classification tasks.

```
# Plot the image with the most likely prediction
plt.imshow(img)
plt.axis('off')
plt.title(f"Prediction: {top_classes[0]} ({top_confidences[0]:.2f}%)")
plt.show()
```

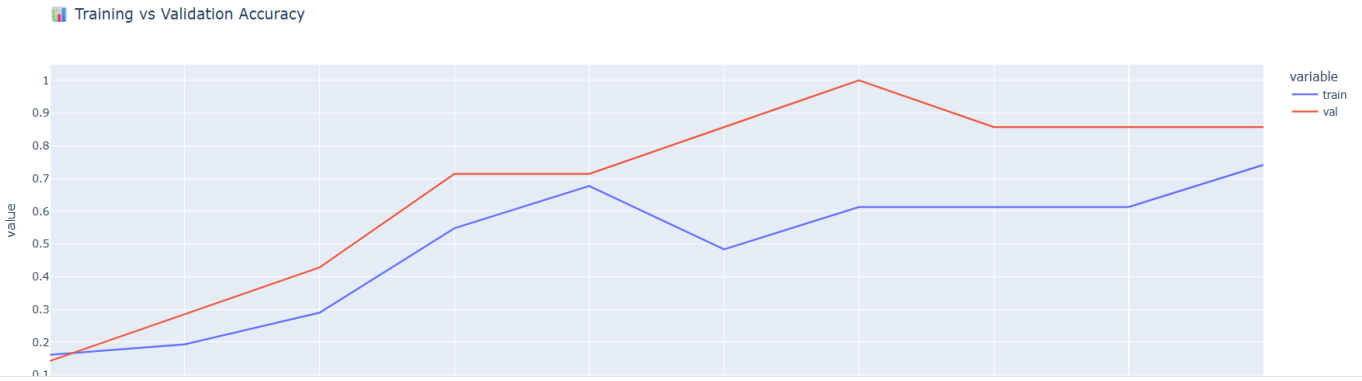




```python
import pandas as pd
import plotly.express as px

acc = pd.DataFrame({
    'train': history.history['accuracy'],
    'val': history.history['val_accuracy']
})

fig = px.line(acc, x=acc.index, y=acc.columns[0:], title='📊 Training vs Validation Accuracy')
fig.show()
```



Conclusion

'Grain Palette' demonstrates how deep learning and web technology can come together to solve real-world agricultural problems efficiently. Future work may include real-time detection, mobile deployment, and adding more varieties.

Reference

1. TensorFlow Documentation
2. Kaiming He et al. (ResNet Paper)
3. MobileNetV2: https://arxiv.org/abs/1801.04381
4. Flask Documentation
5. Murat Koilu Dataset Source