PROJECT TITLE

FRUIT AND VEGETABLE IMAGE RECOGNITION

Introduction

In an era characterized by increasing digitalization and technological advancements, image recognition has emerged as a pivotal field with diverse applications across various industries. One such application is the recognition and classification of fruits and vegetables, which plays a crucial role in agriculture, retail, food processing, and dietary assessment.

The "Fruit and Vegetable Image Recognition" project aims to leverage deep learning techniques to develop a robust model capable of accurately identifying and categorizing different types of fruits and vegetables from images. By harnessing the power of convolutional neural networks (CNNs) and advanced machine learning algorithms, this project seeks to automate the process of fruit and vegetable classification, thereby enhancing efficiency, productivity, and decision-making in relevant domains.

1. Project Overview

The goal of this project is to develop a robust deep learning model capable of accurately identifying and classifying various fruits and vegetables from images. The dataset used for training and validation comprises 36 different classes of fruits and vegetables, making this a multi-class classification problem.

2. Dataset Acquisition and Pre-processing

2.1 Kaggle Dataset Download:

Dataset link - https://www.kaggle.com/datasets/kritikseth/fruit-and-vegetable-image-recognition

- Data is collected from Kaggle open-source website
- Utilizing Kaggle API for data acquisition
- Kaggle.json file is configured and used for authentication with Kaggle API

2.2 About the Data

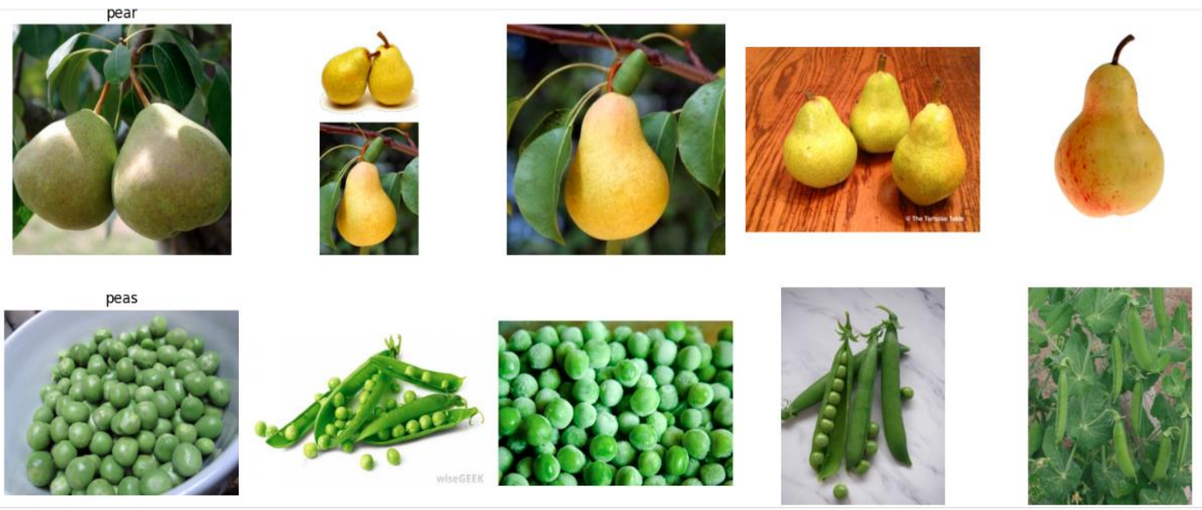
This dataset encompasses images of various fruits and vegetables, providing a 36 collection of classes for image recognition tasks. The included food items are:

- **Fruits:** Banana, Apple, Pear, Grapes, Orange, Kiwi, Watermelon, Pomegranate, Pineapple, Mango
- **Vegetables:** Cucumber, Carrot, Capsicum, Onion, Potato, Lemon, Tomato, Radish, Beetroot, Cabbage, Lettuce, Spinach, Soybean, Cauliflower, Bell Pepper, Chilli Pepper, Turnip, Corn, Sweetcorn, Sweet Potato, Paprika, Jalapeño, Ginger, Garlic, Peas, Eggplant

The dataset is organized into three main folders:

- **Train:** Contains 100 images per category.
- **Test**: Contains 10 images per category.
- **Validation**: Contains 10 images per category.

2.3 How the Data Looks Like



2.2 Data Pre-processing

The dataset is organized into training and validation sets using the tf.keras.utils.image_dataset_from_directory function.

Images are resized to a uniform size of 64x64 pixels and converted to RGB color mode to ensure consistency.

Data augmentation techniques like shuffling and interpolation are applied to enhance the model's generalization ability.

Labels are inferred from the subdirectories, and categorical encoding is used for labeling.

3. Model Development

We utilized the Keras library for its simplicity and powerful tools to create a sequential model

3.1 Initial Model:

1. Convolutional Layers:

- The model starts with two convolutional layers, each with 64 filters and a kernel size of 3x3. These layers are responsible for extracting features from the input images.
- After each convolutional layer, a Max Pooling layer is added with a pool size of 2x2 and strides of 2. This reduces the spatial dimensions of the feature maps and helps in downsampling.

2. Dropout Layer:

- A Dropout layer with a rate of 0.5 is included to prevent overfitting. It randomly sets 50% of the input units to 0 during each forward pass during training, which helps the model generalize better.

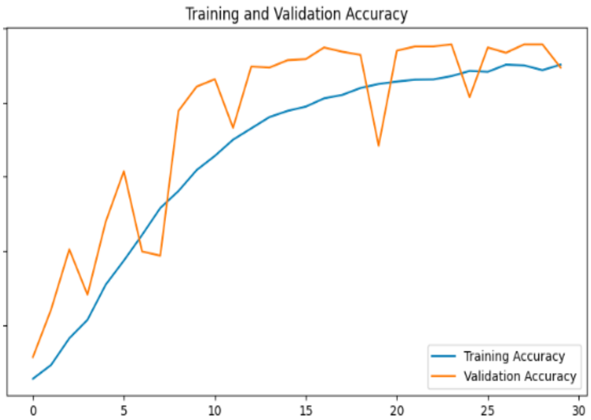
3. Flatten Layer:

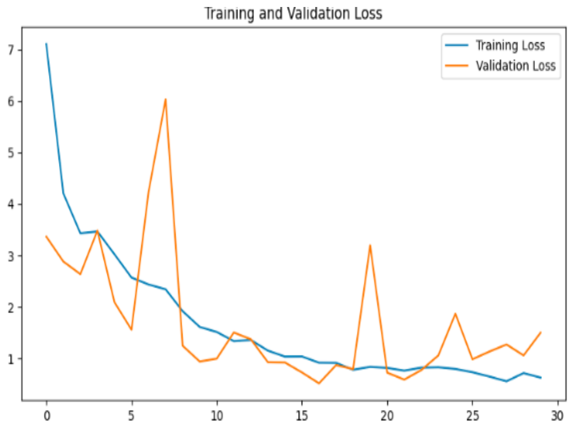
- The Flatten layer converts the 2D feature maps obtained from the convolutional layers into a 1D array. This flattened array is then fed into the fully connected (dense) layers.

4. Dense (Fully Connected) Layers:

- The model includes a dense layer with 128 units and a ReLU activation function. This layer learns to interpret the high-level features extracted by the convolutional layers.
- The final dense layer has 36 units and a softmax activation function. This layer outputs a probability distribution over the 36 classes, providing the classification results for the input images.

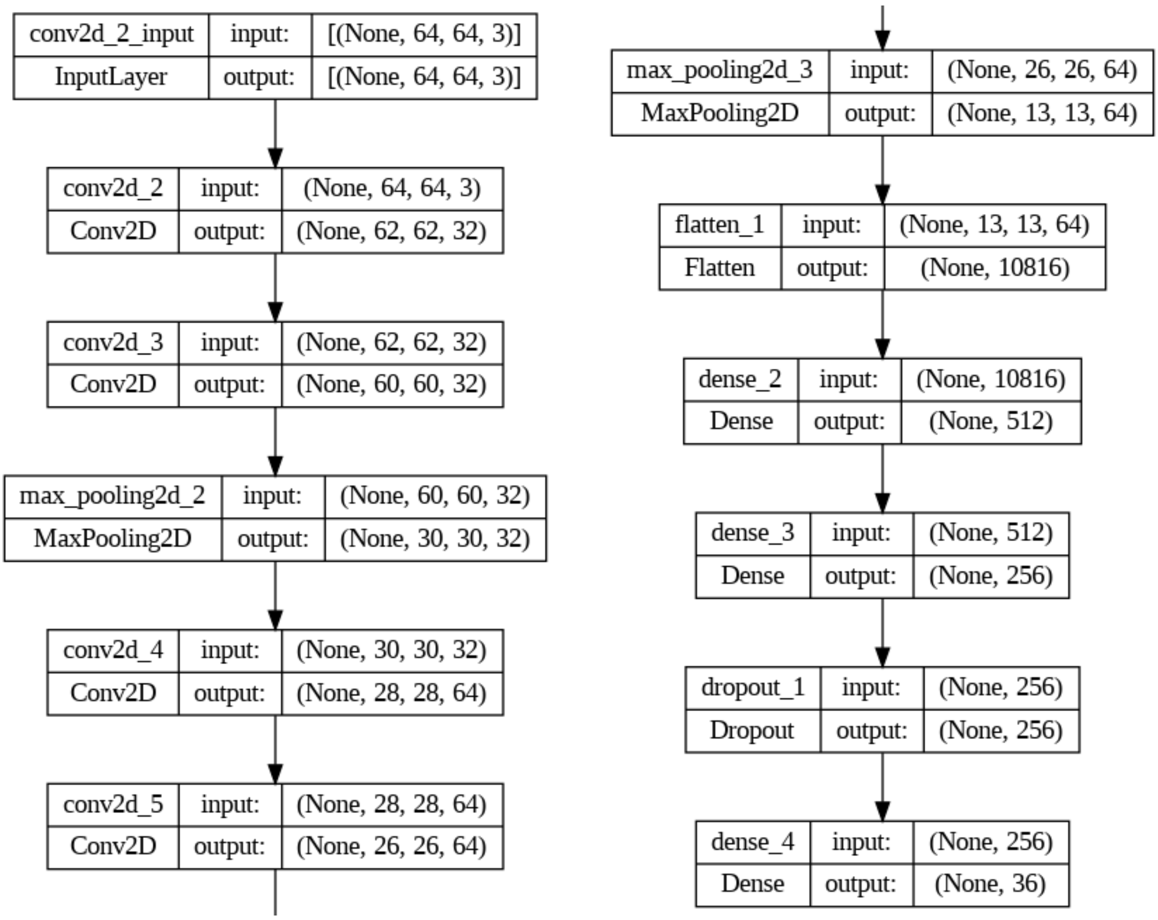
3.2 Model's Performance:





- As we can see the model is fluctuating to converge and showed problem of underfitting
- Validation accuracy and validation loss jumped since the start and never converged
- To tackle this issue we have increased the model's complexity and changed the optimizer to Adam for smooth convergence.

3.3 Improving Model's Performance



Second model is created with four convolutional layers two of 32 filters each another two with 64 filters each and a kernel of 3 because we have coloured images. With an activation function of relu in each layer.

Maxpooled every other two conv layers to exctract only important features from the images, flattened the output tensor into 1D array to feed the Artificial Neural Network

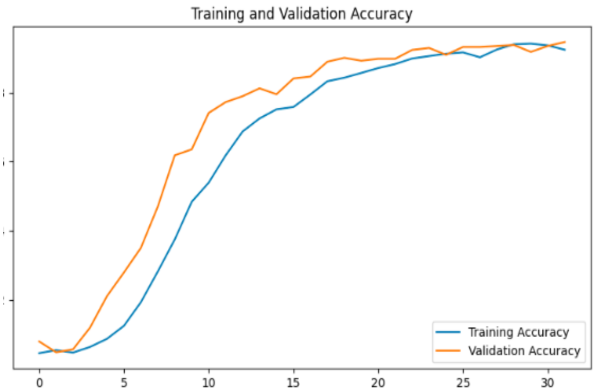
Created first dense layer of 512 nodes and an activation function of relu, second dense layer of 256 nodes to create little complexity in the model. Added a dropout layer for turning og 50% neuron to avoid overfitting.

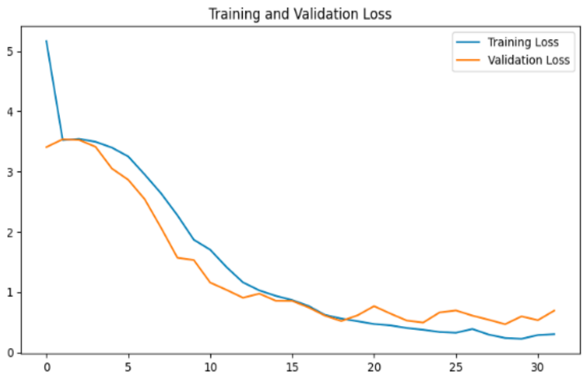
Finally created an output layer with 36 neurons for 36 classes from training data. Utilized softmax activation function.

3.4 Model's Performance

- The initial model is trained for 30 epochs, and training metrics such as accuracy and loss are monitored.

- After model improvement, the enhanced model is trained for 32 epochs to further refine its performance.





Model converged after 25 epochs, returning a better results than previous model. This model has'nt shown underfitting and bounces while convergence.

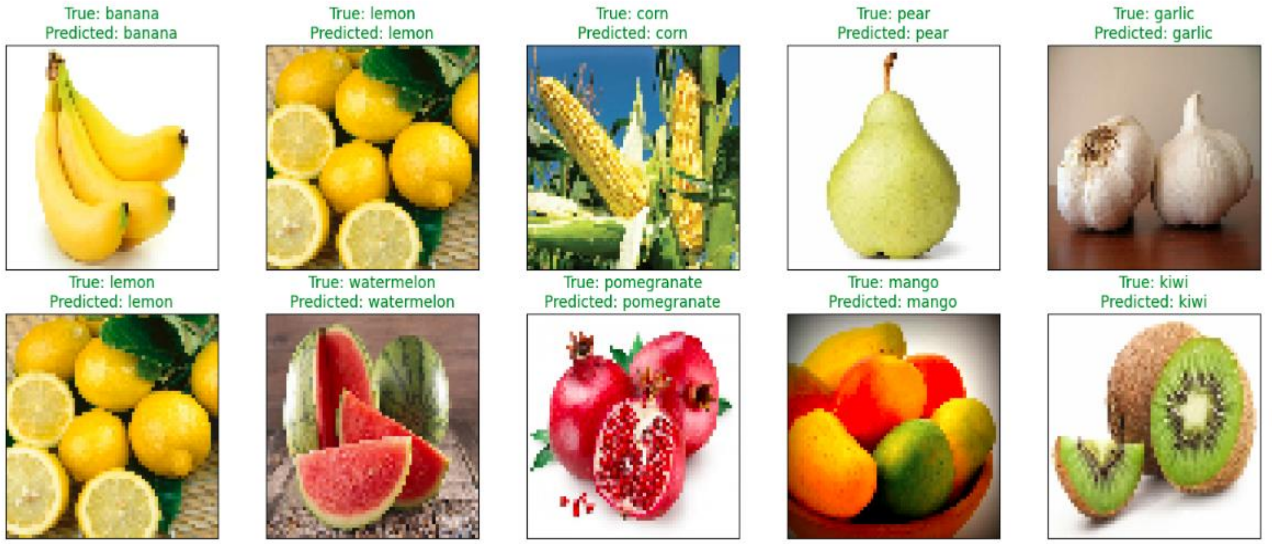
3.5 Model Evaluation

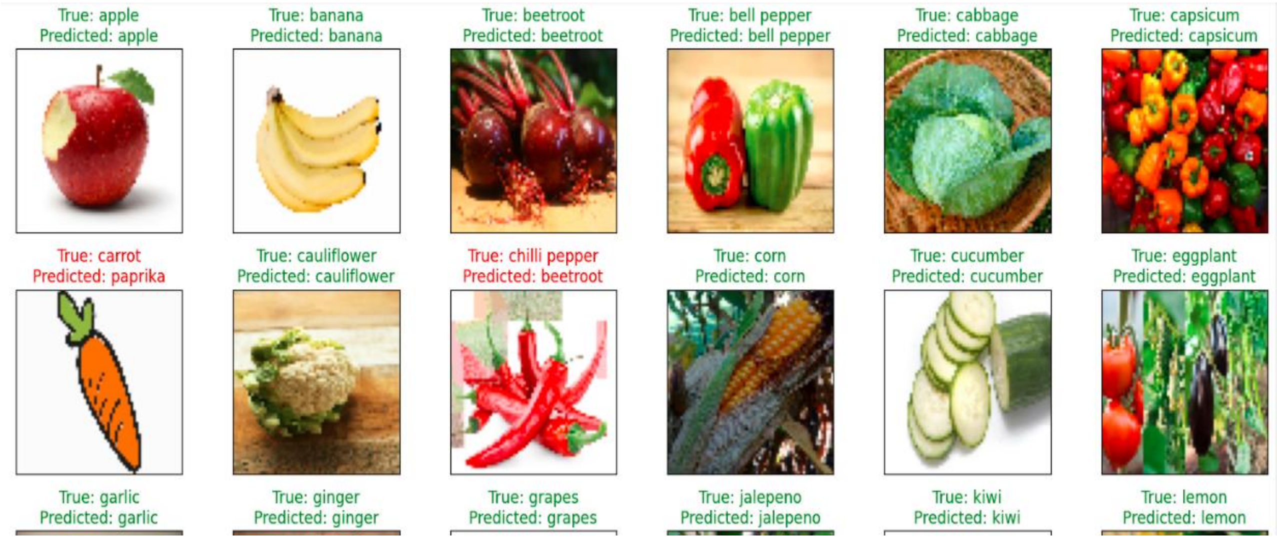
- Training and validation accuracies and losses are evaluated using the model.evaluate method to assess model performance.
- A classification report is generated to analyse precision, recall, and F1-score metrics for each class, providing detailed insights into the model's classification capabilities.

3.6 Model Results

The final model achieves an impressive accuracy of **95%** on the validation set, demonstrating its effectiveness in accurately classifying fruits and vegetables from images.

4. Model Predictions





CONCLUSION

The detailed documentation and analysis provided throughout the project enable a thorough understanding of the data, model development process, training, evaluation, and key performance metrics. Further optimizations and fine-tuning strategies can be explored to enhance the model's accuracy, robustness, and real-world applicability.

This comprehensive documentation encapsulates the entire project lifecycle, from data acquisition and preprocessing to model development, training, evaluation, and result analysis. It serves as a valuable resource for understanding the project's methodology, challenges faced, insights gained, and areas for potential future enhancements and research.