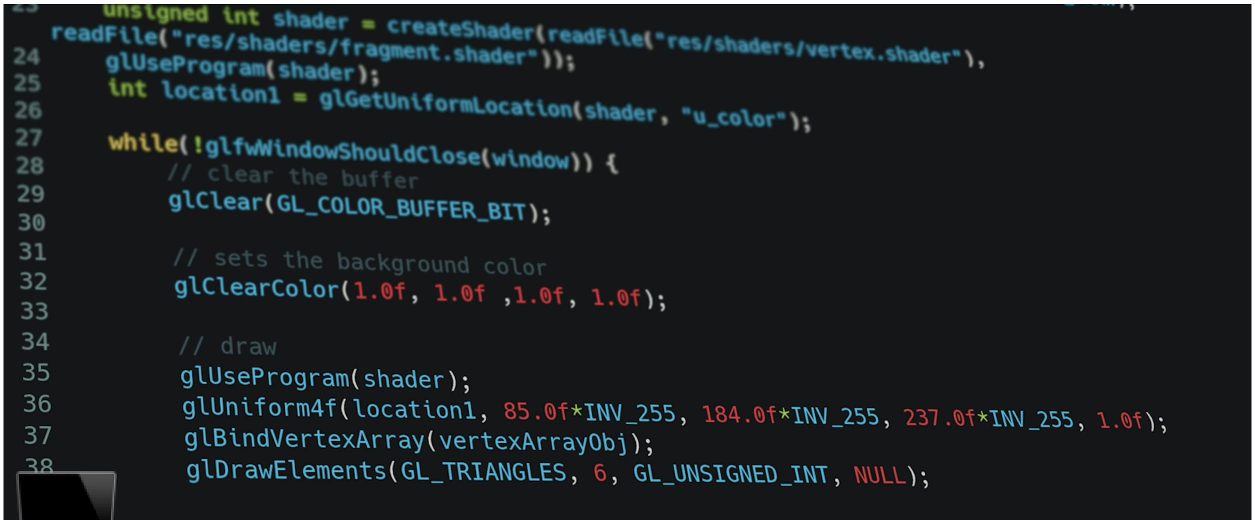

Fruit and Vegetable Image Classification Project Documentation

Students

Name	Level	ID
Yassen Al-Sayed Taalab	3	15210369
Mahmoud Reda Elmetwally	3	15210307
Mahmoud Abdelmaqsoud kroush	3	15210313

Project Overview

This project focuses on building a deep learning-based image classification system to identify fruits and vegetables from images. The model is trained using TensorFlow and Keras on a dataset containing images of various fruits and vegetables, aiming to achieve high accuracy in classification.

Objectives

1. To design a robust image classification system using convolutional neural networks (CNNs).
2. To preprocess and visualize a real-world dataset for effective training.
3. To evaluate and optimize the model's performance.
4. To create a user-friendly prediction system for single image classification.

Dataset

The project uses a custom dataset stored in the following directory structure:

```
Fruits_Vegetables/
    train/
    test/
    validation/
```

Each folder contains images categorized by their respective labels, e.g., apples, bananas, carrots, etc. The dataset is split into training, validation, and test sets to ensure proper evaluation of the model.

Methodology

1. Data Preprocessing

- **Image resizing**: All images are resized to 180×180 pixels.
- **Batching**: The images are grouped into batches of size 32 for training.
- **Rescaling**: Pixel values are normalized to the range [0, 1] using the `Rescaling` layer.

2. Model Architecture

The CNN model is built using the TensorFlow Keras Sequential API with the following layers:

- **Rescaling Layer**: Normalizes pixel values.
- **Convolutional Layers**: Extract features using filters of size 3×3 with ReLU activation.

- **MaxPooling Layers**: Reduces spatial dimensions to focus on dominant features.
- **Flatten Layer**: Converts the 2D output into a 1D vector.
- **Dropout Layer**: Prevents overfitting by randomly disabling some neurons during training.
- **Dense Layers**: Fully connected layers for classification, ending with an output layer matching the number of categories.

3. Training

- **Optimizer**: Adam optimizer for efficient gradient descent.
- **Loss Function**: Sparse categorical crossentropy to handle multi-class classification.
- **Metrics**: Accuracy to evaluate model performance.
- **Epochs**: Trained for 25 epochs using training and validation datasets.

4. Visualization

- Training and validation accuracy and loss are plotted over epochs to observe learning trends and detect overfitting or underfitting.

5. Prediction

- Single image predictions are implemented using:
  - Image loading and resizing.
  - Conversion to a batch tensor.
  - Applying the trained model to predict the class with the highest confidence.

Results

- **Accuracy**: The model achieved high accuracy on both training and validation datasets, as seen in the plotted metrics.
- **Loss**: Validation loss remained stable, indicating good generalization.

Directory Structure

```
project-root/
    ├── Fruits_Vegetables/        # Dataset directory
    ├── Image_Class_Model.ipynb   # Jupyter notebook with the
code
    ├── requirements.txt        # Required Python packages
    ├── saved_model/            # Directory for the trained m
odel
    └── README.md               # Project overview and instru
ctions
```

Future Work

1. **Expand Dataset**: Include more categories and images to improve generalization.
2. **Real-Time Predictions**: Integrate a real-time webcam prediction feature.
3. **Model Optimization**: Explore advanced techniques such as transfer learning or hyperparameter tuning.
4. **Deployment**: Deploy the model as a web application using Flask or Django.

Conclusion

This project demonstrates the implementation of a CNN for image classification of fruits and vegetables. It covers all essential steps from data preprocessing to model training and prediction, providing a strong foundation for further development and deployment.