Fruit Detector Using MobileNetV2 and Webcam

1. Project Description

This code implements real-time fruit detection using a webcam and a deep learning model built on the **MobileNetV2** architecture. It includes:

- Training the model with a dataset of various fruits.
- Using the webcam to detect the type of fruit based on captured images.

---

2. Code Explanation

i) Importing Libraries

- **Os** : Handles file and directory operations.
- **Numpy** : For numerical array manipulation.
- **matplotlib.pyplot** : Visualizes training results (accuracy and loss).
- **Tensorflow** : A framework for deep learning.
- **cv2 (OpenCV)** : Captures and processes images from the webcam.
- **tensorflow.keras** : Provides tools for building and training the neural network.

ii) Data Augmentation

- **Purpose** : Increases the diversity of training data, helping the model generalize better.
- **Code** :

```
train_datagen = ImageDataGenerator(
  rescale=1.0 / 255,
  rotation_range=20,
  width_shift_range=0.2,
  height_shift_range=0.2,
  shear_range=0.2,
  zoom_range=0.2,
  horizontal_flip=True
)
```

```
test_datagen = ImageDataGenerator(rescale=1.0 / 255)
```

- **Explanation**:

  (a) **Rescale** : Normalizes pixel values from [0, 255] to [0, 1].

  (b) **Data augmentation in training** : Applies random rotations, shifts, zoom, and horizontal flipping.

  (c) **Testing data** : Only rescales images to preserve dataset integrity.

iii) Loading the Dataset

The dataset is structured into folders based on fruit classes. For example:

```
o   train/
        ├– apple/
        ├– banana/
        ├– mango/
o   test/
        ├– apple/
        ├– banana/
        ├– mango/
```

- **Code** :

```
train_data = train_datagen.flow_from_directory(
   r"PATH_TO_TRAIN_DIRECTORY",
   target_size=(224, 224),
   batch_size=32,
   class_mode='categorical',
   color_mode='rgb'
)
test_data = test_datagen.flow_from_directory(
   r"PATH_TO_TEST_DIRECTORY",
   target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    color_mode='rgb'
)
```

iv) Transfer Learning Model

- **Architecture** : MobileNetV2 is used as the base model with pre-trained weights from **ImageNet**.
- **Customization**:
  a) Additional layers such as **Flatten**, **Dropout**, and **Dense** are added for fruit classification (10 classes).
- **Code** :

```
base_model = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
base_model.trainable = False


model = Sequential([
  base_model,
  Flatten(),
  Dropout(0.25),
  Dense(128, activation='relu'),
  Dense(10, activation='softmax')
])
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

v) Training the Model

- **Process** :
  a) The model is trained on the training data for 15 epochs.

  b) Validation is performed on the test data after each epoch.
- **Code** :

```
history = model.fit(
    train_data,
    epochs=15,
    validation_data=test_data
)
```

vi) Model Evaluation

- The model is evaluated on the test dataset to determine its accuracy.
- **Code** :

```
test_loss, test_accuracy = model.evaluate(test_data)
print(f"Test Accuracy: {test_accuracy:.2f}")
```

- **Output**: Displays the model's accuracy on the test data.

vii) Saving the Model

The trained model is saved in the **.h5** format, so it can be used later without retraining.

- **Code** :

```
model.save('fruit_classifier_model.h5')
```

viii) Real-Time Fruit Detection using Webcam

- **Steps** :
  a) Open the webcam using **OpenCV**.
  b) Resize the captured frame to **224x224**.
  c) Predict the class using the trained model.
  d) Display the predicted label on the webcam feed.
- **Code** :

```
cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()
    if not ret:
        break

    img = cv2.resize(frame, (224, 224))
    img = img_to_array(img) / 255.0
    img = np.expand_dims(img, axis=0)
    predictions = model.predict(img)
    class_index = np.argmax(predictions)
    label = labels[class_index]

    cv2.putText(frame, f"It matches with: {label}", (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)
    cv2.imshow("Fruit Detector by Safwan & Arsyi", frame)
    if cv2.waitKey(1) & 0xFF == ord('x'):
        break
cap.release()
cv2.destroyAllWindows()
```

3. Key Features

- **Transfer Learning** : MobileNetV2 is used as a base model, saving training time and resources.
- **Data Augmentation** : Helps the model generalize better.
- **Real-Time Webcam Detection** : Detects fruit types directly from webcam input.
- **High Accuracy** : Achieves high accuracy thanks to transfer learning.

4. How to Run the Code

1. **Dataset** :

Prepare the dataset in a directory structure like this:

- train/

├– apple/

├– banana/

- test/

  ├– apple/

  ├– banana/

Ensure the images in the folders are in formats such as .jpg or .png.

2. **Run the Training Code** :
   - Adjust the dataset paths in train_datagen and test_datagen.
   - Execute the training code.
3. **Using the Webcam :**
   - Make sure the webcam is conenected
   - Run the webcam detection code
   - Press ‘x’ to exit the application

---

5. Notes

- Ensure **TensorFlow** and **OpenCV** are installed in your Python environment.
- The model can be further optimized by fine-tuning (unfreezing some layers of MobileNetV2).

Contributing

Contributions are welcome! You can fork the repository, make changes, and submit a pull request. Suggestions for improving accuracy, adding new features, or optimizing the code are appreciated.