TASK 1
FRUIT CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS AND TRANSFER LEARNING

ARNAV GUPTA – RA2211028010107

SPANDAN BASU CHAUDHURI – RA2211028010120

AIM

The aim of this project is to develop a robust image classification system capable of identifying different types of fruits (Apple, Banana, and Orange) from images using deep learning models. The project compares a basic Convolutional Neural Network (CNN) with a Transfer Learning model based on MobileNetV2 to analyze performance differences in accuracy and efficiency.

OBJECTIVES

1. To download, preprocess, and prepare the fruit image dataset for machine learning tasks.
2. To extract classification labels from the dataset's XML annotations.
3. To construct and train a custom CNN model from scratch for fruit classification.
4. To implement Transfer Learning using MobileNetV2 and train the model on the same dataset.
5. To fine-tune the pretrained network for improved performance.
6. To compare model performance using metrics such as accuracy, confusion matrix, and classification report.
7. To save and deploy the best-performing trained model for real-time inference.

INTRODUCTION

Image classification is a key problem in the field of computer vision, where the objective is to identify the category or class to which an image belongs. With the advent of Deep Learning, Convolutional Neural Networks (CNNs) have become the standard approach for handling image-related tasks due to their ability to automatically learn spatial hierarchies and visual patterns.

In this project, we work with a dataset that contains images of fruits (Apple, Banana, and Orange), each accompanied by annotation files meant originally for object detection. However, instead of detecting objects, we simplify the problem by extracting class labels for classification only.

Two approaches are used:

- A **Simple CNN Model**, built and trained from scratch.
- A **Transfer Learning Model** using pretrained **MobileNetV2**, a lightweight but powerful neural network originally trained on ImageNet.

Since MobileNetV2 already has feature extraction knowledge, it performs better and requires less data and training time. Further fine-tuning of selected layers helps enhance accuracy for our specific dataset.

REQUIREMENTS

Hardware Requirements:

- A computer with at least 8GB RAM
- GPU recommended (e.g., NVIDIA CUDA GPU) for faster training

Software Requirements:

Software	Version
Python	3.8+
Jupyter Notebook or Google Colab	Optional
TensorFlow	2.x
KaggleHub	latest
Matplotlib, NumPy, PIL, sklearn	latest

Dataset:

- **Kaggle Dataset:** mbkinaci/fruit-images-for-object-detection

The dataset consists of structured directories with images and XML annotation files.

LIBRARIES USED

Library	Purpose
TensorFlow / Keras	Building and training CNN & Transfer Learning models

Matplotlib	Visualizing images and training results
NumPy	Handling arrays and numerical operations
KaggleHub	Automatically download datasets from Kaggle
Pillow (PIL)	Image loading and resizing
xml.etree.ElementTree	Parsing XML annotation files to extract labels
sklearn.metrics	Generating confusion matrix & classification report

ALGORITHM / METHODOLOGY

The steps followed in the experiment are:

- **Dataset Download and Loading**
  The dataset is downloaded via kagglehub.dataset_download() and organized into training and testing directories.
- **Annotation Parsing**
  XML files are parsed to extract fruit labels using ElementTree. Each object label is converted to an index value (0 = apple, 1 = banana, 2 = orange).
- **Image Preprocessing**
  - Images are resized to **224 × 224** pixels.
  - Pixel values are normalized to the range **[0,1]**.
  - Data pipelines are created using tf.data.Dataset.
- **Model Development**
  - **Simple CNN:** Consists of convolution, pooling, flattening, and dense layers.
  - **Transfer Learning Model:** MobileNetV2 base network is used with the top layers replaced.
- **Training**
  - Simple CNN is trained for multiple epochs.
  - Transfer Learning model is trained in two phases:
    1. Freezing base layers (feature extractor stage)
    2. Fine-tuning selected layers for improved learning
- **Evaluation**
  - Accuracy and loss curves are analyzed.
  - Confusion matrix and classification report are generated.

- **Model Saving**
    - The best-performing model is saved as **best_fruit_classifier.h5** for reuse.

INPUT / OUTPUT

Input:

- A color image of a fruit in JPEG format.

Output:

- The predicted fruit category (Apple / Banana / Orange)
- Confidence score of prediction.

Example:

Input Image → Resized → Model Prediction → Output: Banana (97.52%)

Testing: banana_77.jpg

1/1 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ **1s** 1s/step

Prediction: banana (99.84%)

RESULTS

After training and evaluation, the model performance was observed as follows:

Model	Performance Summary
Simple CNN (from scratch)	Moderate accuracy; slower convergence; performance depends strongly on dataset size
Transfer Learning (MobileNetV2)	Significantly higher accuracy; learned general features help classification
Transfer Learning + Fine-tuning	**Best performance**, high precision and recall for all classes

- The final test accuracy after fine-tuning was observed to be **above 93–97%**.
- The classification report indicated strong per-class performance.
- The confusion matrix showed fewer misclassifications after fine-tuning.

This clearly demonstrates the effectiveness of Transfer Learning in real-world datasets with limited size.

Classification Report:

Class	Precision	Recall	F1-Score	Support
Apple	1.00	0.95	0.97	20
Banana	0.95	0.95	0.95	19
Orange	0.95	1.00	0.98	21
Overall Accuracy	**-**	**-**	**0.97**	**60**
Macro Avg	0.97	0.97	0.97	60
Weighted Avg	0.97	0.97	0.97	60

CONCLUSION

This project successfully implemented and compared two different deep learning approaches for fruit image classification. The custom CNN model served as a baseline, while the Transfer Learning approach significantly improved accuracy and training efficiency. Fine-tuning further enhanced the performance, showing that pretrained models generalize well even on smaller datasets.

The trained model can be deployed for:

- Fruit Quality Control Systems in agriculture supply chains,
- Automated fruit sorting in supermarkets,

- Real-time recognition systems in mobile applications using TensorFlow Lite.

In conclusion, Transfer Learning proves to be a powerful technique for achieving high accuracy with minimal computational resources and training data.