Date	21/06/25
Team ID	SWTID1750180744
Project Title	Smart Sorting: Transfer Learning for Identifying Rotten Fruits and Vegetables
Maximum Marks	10 Marks

Preprocessing

The images will be preprocessed by resizing , normalizing ,augmenting,denoising ,adjusting contrast, detecting edges,converting colour space ,cropping batch normalization and whitening data .These steps will enhance the quality , promote model generalization and improve convergence during neural network training ,enduring robust and efficient performance across various computer vision tasks

Section	Description
Data Overview	• dataset:https://www.kaggle.com/datasets/muhammad0subhan/fruit-and-vegetable-disease-healthy-vs-rotten(From kaggle) • The Fruit and Vegetable Diseases Dataset directory contains a comprehensive collection of images categorized to assist in the development of machine learning models for detecting diseases in various fruits and vegetables. This dataset is ideal for tasks such as image classification and deep learning. • The Fruit and Vegetable Diseases Dataset consists of 28 directories, each representing a combination of healthy and rotten images for 14 different types of fruits and vegetables. This

	dataset is intended for training and evaluating machine learning models for disease detection. Images are categorized and organized to facilitate easy access and utilization for image classification and deep learning tasks.
Resizing	Image resizing transforms all input images to a fixed, model-compatible size of 224×224 pixels, enabling reliable, efficient, and consistent training with pre-trained CNN architectures like VGG16. This step is essential for achieving high accuracy in image classification tasks like detecting rotten vs. healthy fruits and vegetables.
Normalization	Normalization in this project rescales all image pixel values from `[0, 255]` to `[0, 1]` to improve training speed, ensure model compatibility, and promote consistent learning across all classes. It is a simple yet crucial step for the success of transfer learning with deep CNNs.
Data Augmentation	Data augmentation in this project increased the diversity of the training dataset by applying random transformations to the images. It enabled better generalization, improved model accuracy, and simulated real-world variation in fruit and vegetable appearance — all without needing to collect more data.
Denoising	• The dataset is already clean. • VGG16 is robust to small noise. • You apply proper augmentation and normalization. • Manual denoising might unintentionally erase key visual cues needed to distinguish rotten from healthy produce.
Edge detection	The model is not explicitly performing edge detection in your code. However, edge detection is happening internally in the convolutional layers of VGG16, which learns to detect edges and shapes as part of its feature extraction process.
Colour space conversion	Colour space conversion is not explicitly used in the project. The model uses the default RGB colour space, which aligns with the VGG16 model's expectations. Converting to other color spaces like grayscale or HSV can be useful for custom models or specific visual tasks, but is not necessary when using standard CNN architectures

	trained on colour images.
Image cropping	Image cropping is not used in the project. All images are resized (not cropped) to 224×224 for compatibility with VGG16. Cropping can be optionally introduced to enhance feature focus, but it requires either manual setup or integration with object detection models (like YOLO, Faster R-CNN)
Batch normalization	Your model has very few trainable layers, so adding Batch Normalization() didn't seem urgent .The model is already using pre-trained weights, which learned good internal feature scaling.
Data processing	

Loading Data	`from google.colab import files`
	`files.upload()`
	`!chmod 600 /root/.kaggle/kaggle.json`
	`!mkdir -p ~/.kaggle`
	`!cp kaggle.json ~/.kaggle`
	`!kaggle datasets download -d muhammad0subhan/fruit-and-vegetable-disease-healthy-vs-rotten`

Resizing	`val_test_datagen = ImageDataGenerator(rescale=1./255)`
	
	`train_generator = train_datagen.flow_from_directory(`
	`    train_dir,`
	`    target_size=IMG_SIZE,`
	`    batch_size=32,`
	`    class_mode='categorical'`
	`)`
	
	`val_generator = val_test_datagen.flow_from_directory(`
	`    val_dir,`
	`    target_size=IMG_SIZE,`
	`    batch_size=32,`
	`    class_mode='categorical',`
	`    shuffle=False`
	`)`

Normalization	IMG_SIZE = (224, 224) train_datagen = ImageDataGenerator(rescale=1./255, rotation_range=25, width_shift_range=0.2, height_shift_range=0.2, shear_range=0.2, zoom_range=0.2, horizontal_flip=True, fill_mode='nearest')
Data Augmentation	train_datagen = ImageDataGenerator(rescale=1./255, rotation_range=25, width_shift_range=0.2, height_shift_range=0.2, shear_range=0.2, zoom_range=0.2, horizontal_flip=True, fill_mode='nearest') val_test_datagen = ImageDataGenerator(rescale=1./255) train_generator = train_datagen.flow_from_directory(train_dir, target_size=IMG_SIZE, batch_size=32, class_mode='categorical')
Denoising	-----Not Used------
Edge Dectection	----Not Used-----
Colour space dectection	-----Not Used-----
Colour space Conversion	-----Not Used-----

<table>
<tr><td>Image Cropping</td><td><pre><code>IMG_SIZE = (224, 224)

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=25,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)</code></pre></td></tr>
<tr><td>Batch normalization</td><td>------Not Used-----</td></tr>
</table>