Module name: Computer Vision

Course track: ~~Basic~~/Advanced/~~Applied~~

Facilitator name: Dr. Richa Dhanuka

Session no: 1

Topics covered: Basics of Computer Vision, A project on Face Recognition

Question 1: What is Computer Vision?

Answer: Computer vision enables computers to detect, identify, and recognize visual information where they can replicate the capabilities of human vision by understanding the complex process of human vision. They use various patterns and features to differentiate and identify visual objects.

Question 2: List out various applications of Computer Vision.

Answer:

* Face Recognition
* Handwriting Recognition
* Object Detection
* Visual Editing
* Self-driving cars

Question 3: What are the differences between object detection and image segmentation?

Answer: Object detection involves identifying and locating objects within an image, typically by drawing bounding boxes around them and labeling the object classes. On the other hand, image segmentation involves dividing an image into multiple segments to identify and delineate each object's boundaries at the pixel level. Object detection focuses on locating objects, whereas image segmentation provides more granular information about the shapes and boundaries of objects within an image.

Question 4: List out various challenges in object detection.

Answer: Below are some of the challenges:

1. A specific object can have many variations
2. Class Imbalancing: Limited data for the positive class and abundant data for negative class, since everything else belongs to the negative class
3. A high value for false positives that should be minimized

Question 5: Explain how a Convolution Neural Network (CNN) can be used for image classification.

Answer: In image classification using CNNs, the network is trained on a dataset with labeled images. The network learns to extract features from images in the initial layers and gradually learns more abstract and complex features in deeper layers. These features are then passed through fully connected layers to make predictions about the image's class. During training, the network adjusts its parameters using optimization algorithms like gradient descent to minimize the difference between predicted and actual classes.

Question 6: How does Transfer Learning work in Computer Vision?

Answer**:** Transfer Learning involves leveraging pre-trained models on large datasets and adapting them to new, smaller datasets or different tasks. In Computer Vision, this means taking a pre-trained model, removing the final classification layers, and adding new layers suited to the specific task or dataset. By using the already learned features from the pre-trained model, transfer learning significantly reduces the amount of data required for training and can boost the performance of models in various Computer Vision tasks.

Question 7: Based on the below code snippet, answer the questions:



What is the purpose of the code snippet provided?

Answer**:** The code captures video frames from a webcam using OpenCV's VideoCapture. It detects faces in the captured frames using a face detection classifier. When a face is detected, it crops the face region and saves it as an image. It displays the original frame with a bounding box around the detected face and a counter indicating the number of detected faces. The loop continues until the 'q' key is pressed or 100 faces are captured.

Question 8: Write a code to capture videoframes using webcam in opencv.

Answer:

Question 9: Explain the purpose of the if-else block starting from **if face\_portion is ():.**

Answer**:** This block checks if any face is detected in the captured frame. If no face is detected (face\_portion is an empty tuple), it prints a message indicating that a face is not found. Otherwise, it extracts the coordinates of the detected face region from the face\_portion list and crops the image around the detected face.

Question 10: Write an exit condition for the while loop, loop should exit when the counter reaches **10** or **k** key is pressed.

Answer:

Session no: 2

Topics covered: Face Recognition System

Question 1: What is a face recognition system, and how does it work?

Answer**:** A face recognition system is a technology that identifies or verifies individuals by analyzing and comparing patterns in facial features from a digital image or video frames. It typically involves several steps: face detection, feature extraction (capturing unique facial characteristics), and matching these features against a database of known. Upon a successful match, the system recognizes the person.

Question 2: What are the key challenges faced by face recognition systems?

Answer**:** Face recognition systems encounter challenges such as variations in lighting conditions, facial expressions, pose variations (tilting or turning of the face), occlusions (covering part of the face), aging, and issues related to the quality of images (blurry, low resolution). Additionally, ensuring accuracy while handling a large database of faces and maintaining privacy and security are crucial challenges.

Question 3: How is transfer learning utilized with VGG16?

Answer**:** Transfer learning involves using pre-trained models on large datasets for tasks with limited data. With VGG16, the model is trained on ImageNet, a vast dataset with millions of labeled images. By leveraging the learned features from ImageNet, one can remove the final classification layers of VGG16 and replace them with new layers for a specific task. Fine-tuning or retraining these modified layers with a smaller dataset helps in achieving good performance even with limited data.

Question 4: How can you load the pre-trained VGG16 model in Python using Keras?

Answer:



Question 5: How can you load pre-trained VGG16 to use as a transfer learning in your own face recognition system?

Answer:

Question 6: Fill in the below code snippet with correct functions (Function1, Function2) and parameters (Para1, Para2).



Answer: 

Question 7: Can you fine-tune the VGG16 model's layers during transfer learning?

Answer: Yes.

Question 8: Write a python code to save and load the model.

Answer:

Question 9: Which keras function is used to do the prediction on the test data?

Answer: predict

Question 10: In machine learning, the test dataset is used to evaluate the \_\_\_\_\_\_\_\_\_\_\_ of a trained model on \_\_\_\_\_\_\_\_\_\_\_ data that the model has not previously seen during the training phase. It serves as a benchmark to assess the model's \_\_\_\_\_\_\_\_\_\_\_ and \_\_\_\_\_\_\_\_\_\_\_ on new instances. Proper separation of data into training, validation, and test sets helps prevent \_\_\_\_\_\_\_\_\_\_\_ and ensures the model's ability to generalize to \_\_\_\_\_\_\_\_\_\_\_ data.

Answer: In machine learning, the test dataset is used to evaluate the **performance** of a trained model on **unseen** data that the model has not previously seen during the training phase. It serves as a benchmark to assess the model's **generalization** and **accuracy** in new instances. Proper separation of data into training, validation, and test sets helps prevent **overfitting** and ensures the model's ability to generalize to **unseen** data.

Session no: 3

Topics covered: Object Detection System

Question 1: What is object detection in the context of computer vision and how does object detection differ from other computer vision tasks?

Answer:Object detection is a computer vision technique that involves identifying and locating objects within an image or video frame.

Object detection not only identifies objects within an image but also provides information about their location by drawing bounding boxes around them.

Question 2: What are some common techniques used in object detection systems?

Answer: Popular techniques include RCNN (Region-based Convolutional Neural Networks), Fast R-CNN, Faster R-CNN, YOLO (You Only Look Once), and SSD (Single Shot Multibox Detector).

Question 3: Can you discuss the trade-offs between speed and accuracy in object detection models?

Answer: There's often a trade-off between speed and accuracy in object detection models. Faster models sacrifice some accuracy for speed, while more accurate models tend to be slower due to their complexity. It's crucial to choose a model based on the specific application requirements, considering factors like real-time processing needs, computational resources, and the acceptable level of accuracy. Often, model architectures are optimized to strike a balance between speed and accuracy for specific use cases.

Question 4: What does “coco.names” file consist of?

Answer: The "coco.names" file typically consists of a list of class names or labels used in the COCO (Common Objects in Context) dataset. This dataset is a widely used benchmark for object detection, segmentation, and captioning tasks in computer vision.

The file contains the names of objects or classes that the COCO dataset encompasses. These names often represent various everyday objects, animals, and other entities that appear in images or videos. For instance, classes like "person," "car," "dog," "cat," "bottle," "chair" and many more might be listed in the "coco.names" file, each representing a distinct object category.

Object detection models or algorithms trained on the COCO dataset use this file to associate numeric class IDs with human-readable labels, allowing the model to identify and classify objects in images or video frames by their respective names or classes.

Question 5: Which tensorflow function is responsible for downloading a pre-trained model from github?

Answer: get\_file



Question 6: Which keras function is used to clear the current keras session?

Answer: tf.keras.backend.clear\_session()

Question 7: What does this code do?Answer: In OpenCV, images are often represented in the BGR (Blue-Green-Red) color space by default, while many other libraries and applications use the RGB (Red-Green-Blue) color space. The above code converts an image from the BGR color space to the RGB color space.

Question 8: What is detection confidence score in the object detection system?

Answer: When an object detection algorithm runs on an image, it not only identifies objects but also provides a measure of confidence associated with each detected object. This confidence score indicates the algorithm's belief or certainty that the detected object belongs to a specific class or category and that the bounding box surrounding the object is accurate.

Typically, the confidence score is a value between 0 and 1, where:

0 indicates low confidence or low certainty that the detected object is correctly classified.

1 indicates high confidence or high certainty that the detected object is accurately classified.

Question 9: Write a function called *createBoundingBox* to create a bounding box and display the confidence score for the detected objects.

Answer: 

Question 10: What is fps in terms of video frames?

Answer: FPS stands for "Frames Per Second." It's a measurement used to describe the frame rate or the frequency at which consecutive images (frames) are displayed or processed in a video.

Higher FPS values generally result in smoother and more realistic motion in videos because there are more frames shown per second, reducing the perception of motion blur and choppiness. Common FPS values used in videos include 24 FPS, 30 FPS, 60 FPS, and higher frame rates like 120 FPS or 240 FPS, which are often used in gaming or high-speed video applications.

Session 4

Topics Covered: Image Segmentation

Question 1: What is YOLO, and how does it differ from traditional image segmentation methods?

Answer: YOLO, or You Only Look Once, is an object detection algorithm that has been used to create high-speed image segmentation architectures. Unlike traditional segmentation methods that classify each pixel, YOLO divides the image into a grid and predicts bounding boxes and associated class probabilities for each grid cell, making it faster and more efficient.

Question 2: Could you discuss a real-world application where image segmentation plays a crucial role and the challenges faced in that context?

Answer: In medical imaging, segmenting tumors or organs from scans is critical for diagnosis and treatment planning. Challenges involve accurately delineating complex structures, dealing with variations in image quality due to different imaging devices, and handling the presence of anomalies or noise in scans.

Question 3: Can image segmentation be applied in quality control processes?

Answer: Yes, in industries like manufacturing, segmentation can help inspect products by identifying defects or irregularities in materials or finished goods.

Question 4: What ethical considerations are associated with using image segmentation in AI applications?

Answer: Ethical concerns revolve around privacy (if segmentation involves personal data), biases in algorithms, and the potential misuse of segmentation technology in surveillance or tracking without consent.



Question 5: What is the significance of the *yolov8m-seg.pt* model used in the code snippet?

Answer: The *yolov8m-seg.pt* model represents YOLO version 8 optimized for segmentation tasks. The 'm' in the name refers to the model size, which can vary from 'nano' to 'xl', indicating the model's capacity and complexity. This model aims to perform segmentation, identifying objects and their boundaries within an image.

Question 6: How does the code snippet utilize the YOLO model to perform image segmentation?

Answer: The code snippet uses the Ultralytics library to instantiate a YOLO model for segmentation (*model=YOLO("yolov8m-seg.pt")*). It then predicts segmentations for an image (*predict = model.predict(".\\data\\image1.jpg", save=True, save\_txt=True)*), extracting masks for objects detected in the image.

Question 7: How does the code snippet extract and save segmented masks from the YOLO predictions?

Answer: The code snippet accesses the segmented masks from the predictions (*predict[0].masks*) and converts them into images by reshaping and adjusting pixel intensities (*seg\_image = (predict[0].masks[3].data.numpy().reshape(416, 640) \* 255).astype("uint8")*). These masks are saved as images using OpenCV's *cv2.imwrite()* function.

Question 8: What purpose does the bound.jpg image serve in the code snippet, and how is it generated?

Answer: The *bound.jpg* image represents the boundaries of detected objects. It's generated by creating an image array (*b*) of zeros with the same dimensions as the original image, then marking the predicted object locations by setting corresponding pixel values to *255*.

Question 9: The below script prints the associated class \_\_\_\_\_\_\_\_\_\_\_\_ for each segmented region in the image. Fill in the blanks.



Answer: labels

Question 10: The model used in the code snippet, yolov8m-seg.pt, is a YOLO version optimized specifically for \_\_\_\_\_\_\_\_\_\_\_\_ tasks. Fill in the blanks.

Answer: segmentation

Session 5:

Topics covered: Pose estimation

Question 1: What is pose estimation in computer vision?

Answer: Pose estimation involves determining the position, orientation, and often the movement of objects or people within an image or a video frame. In human pose estimation, it specifically refers to locating and tracking key points or joints on a person's body, such as elbows, shoulders, knees, etc., to understand their pose or stance.

Question 2: What are some common applications of pose estimation?

Answer: Pose estimation finds applications in various domains:

* Human-Computer Interaction: Gesture recognition, sign language translation, and virtual reality.
* Sports Analysis: Analyzing athletes' movements for performance improvement.
* Healthcare: Monitoring patient movements for rehabilitation or analyzing gait for diagnosing certain conditions.
* Surveillance and Security: Tracking people’s movements for security purposes.
* Animation and Gaming: Creating realistic character movements in animation and gaming.

Question 3: What are some challenges in pose estimation?

Answer: Several challenges include:

* Occlusion: Body parts can be hidden or occluded, making it challenging to accurately estimate poses.
* Variability in Pose: People can have different body shapes and sizes, leading to variations in poses.
* Real-Time Processing: Real-time applications demand quick and accurate pose estimation, which poses computational challenges.
* Ambiguity: Some poses might look similar, making it difficult for algorithms to distinguish between them accurately.

Question 4: List out various python packages for pose estimation.

Answer: OpenPose, PoseNet, Ultralytics, MediaPipe, etc.

Question 5: Elaborate on the capabilities of MediaPipe?

Answer: MediaPipe is an open-source framework developed by Google that offers customizable machine-learning solutions for various perception tasks, including pose estimation, face detection, hand tracking, and more. Various capabilities include:

* Good balance between accuracy and efficiency
* Can estimate multiple poses simultaneously in real time
* Supports multiple programming languages
* Cross-platform

Question 6: What keypoints does MediaPipe Pose detect?

Answer: MediaPipe Pose can detect and track various keypoints, including those representing the wrists, elbows, shoulders, hips, knees, and ankles. It identifies these keypoints to reconstruct and understand human body poses.

Question 7: What library is being used for pose estimation in the code snippet?



Answer: The code uses the MediaPipe library for pose estimation. It specifically utilizes mp\_pose from MediaPipe for detecting and tracking key landmarks on a person's body.

Question 8: What does the *mp\_drawing.draw\_landmarks()* function do in the code?

Answer: This function is responsible for drawing the detected pose landmarks and connecting lines between them on the video frame. It visualizes the landmarks and their connections using different colors and drawing specifications.

Question 9: Which function is responsible for drawing the connections on the real-time video feed?

Answer: *mp\_drawing.draw\_landmarks*

Question 10: Extend the code to find the coordinates of the right shoulder.

Answer: *result.pose\_landmarks.landmark[mp\_pose.PoseLandmark.RIGHT\_SHOULDER.value]*