**HUMAN POSE DETECTION**

**PROJECT EXPLANATION :**

**Objective:** The project aims to detect human poses in images. Pose estimation is a fundamental task in computer vision, with applications ranging from sports analytics to gesture recognition.

**Methodology:** The project uses a pre-trained deep learning model for pose estimation. Deep learning, particularly convolutional neural networks (CNNs), has shown significant success in image-based tasks like object detection and pose estimation.

**Model Selection:** The selected model is likely a variant of MobileNet, a lightweight CNN architecture suitable for mobile and embedded devices. This model is trained on a large dataset of human poses.

**Implementation:** The code implements pose estimation by processing an input image through the pre-trained model. The model outputs heatmaps representing the likelihood of each body part's presence. These heatmaps are then used to extract the coordinates of body parts.

**Visualization:** The detected poses are visualized by drawing lines connecting key body parts, forming a skeleton-like structure. This visualization helps understand the detected poses.

**Performance:** The code includes performance profiling to measure the inference time of the model. This information is valuable for assessing the model's efficiency, especially in real-time applications.

Overall, the project demonstrates the use of deep learning for human pose detection

**CODE:**

A screenshot of a computer

Description automatically generated

A computer screen shot of a person wearing a suit

Description automatically generated

A screenshot of a computer

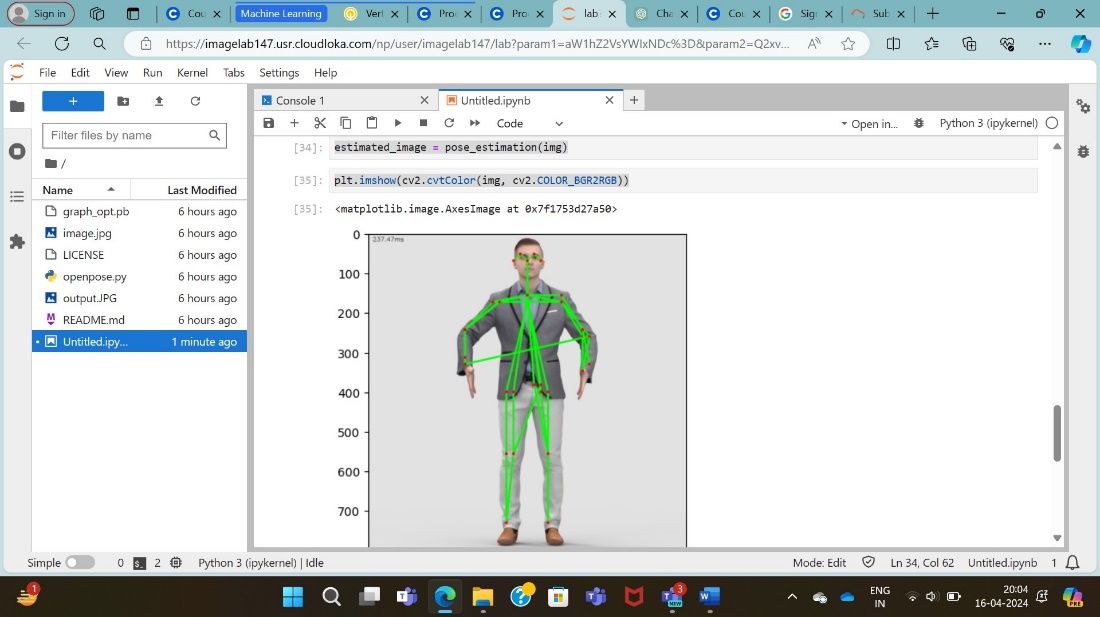
Description automatically generated

A computer screen shot of a computer screen

Description automatically generated

A computer screen shot of a computer screen

Description automatically generated



**CODE EXPLANATION:**

This code performs human pose estimation using OpenCV's DNN module with a pre-trained TensorFlow model.

**Importing Libraries:** The code starts by importing necessary libraries such as OpenCV (cv2) and Matplotlib (plt).

**Loading the Pre-trained Model:** It loads the pre-trained model using cv2.dnn.readNetFromTensorflow. The model is expected to be in TensorFlow's protobuf format (graph\_opt.pb).

**Defining Constants:** Constants like input width, height, and confidence threshold are defined.

**Body Parts and Pose Pairs:** A dictionary BODY\_PARTS is defined which maps body part names to their respective indices in the output of the neural network. POSE\_PAIRS defines pairs of body parts which are used to draw lines connecting body parts.

**Reading Image:** An image (image.jpg) is read using cv2.imread.

**Pose Estimation Function**: pose\_estimation(frame) function is defined which takes a frame (image) as input and performs pose estimation on it. It utilizes the pre-trained neural network to detect body parts' heatmaps and then extracts the coordinates of the body parts.

**Drawing Skeleton:** Within the pose\_estimation function, the detected points are used to draw lines connecting body parts based on predefined POSE\_PAIRS.

**Performance Profiling:** The performance of the neural network is measured and displayed on the output frame.

**Displaying Results:** After pose estimation, the estimated image is displayed using Matplotlib.