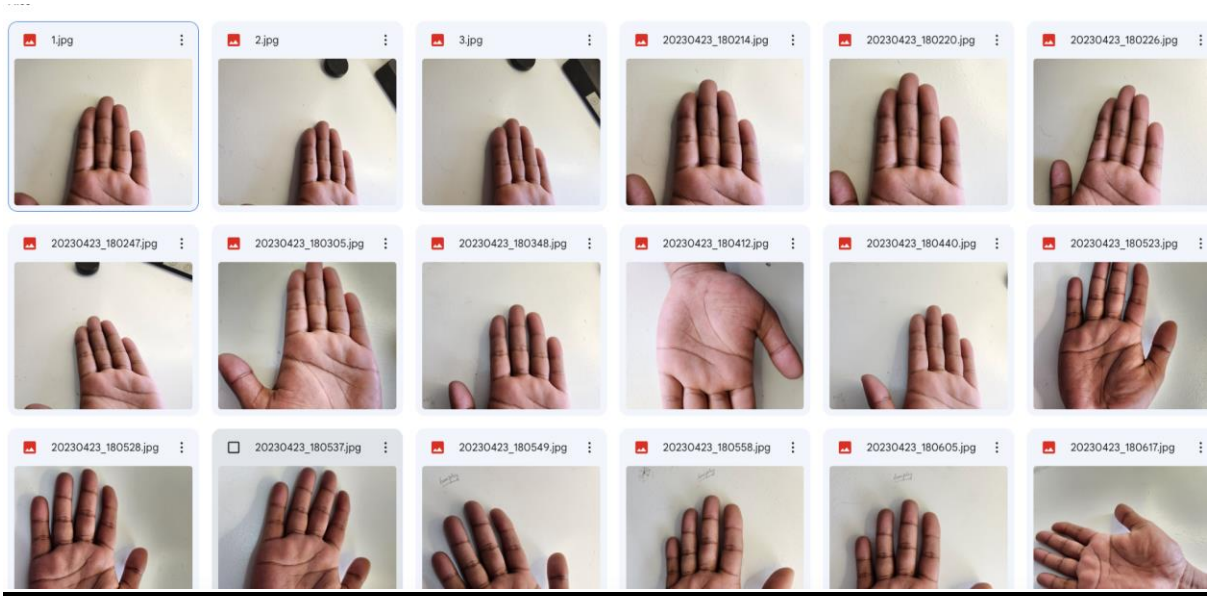
CSE666 Programming Assignment 2

Task1: Dataset collection



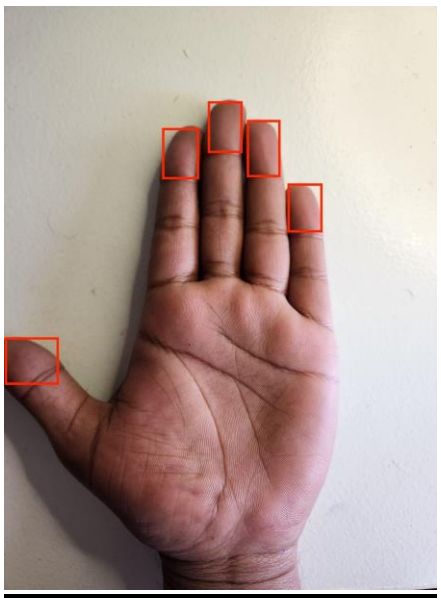
The palm pictures were captured . 10 images for each of the hands.

Task 2: Annotation

Approach

Bounding boxes were marked for each fingertip of the palm images captured

Result



Annotated image

Task 3: Fingerprint Detection

Approach

The code uses TensorFlow, OpenCV, NumPy, and **Mediapipe** libraries to detect hands in images, extract the coordinates of the fingertips of each finger, and draw a bounding box around the fingertips. It initialises the hand detection model using the **Mediapipe** library with some specific configurations.

The code then processes the images using the hand detection model and extracts the results. For each detected hand in the image, the code extracts the pixel coordinates of the hand landmarks, which are the locations of specific points on the hand, and uses these landmarks to extract the coordinates of each finger's fingertips.

For each fingertip, the code creates a bounding box around it and draws the bounding box on the image. Finally, the code saves the image with the drawn bounding boxes to the output results directory.

Results

The code successfully detects fingertips in the images and draws bounding boxes over each fingertip

As we can see from the above image, almost all fingertips were detected, except for thumb.

Code snippet:

```python
import tensorflow as tf
import os
import cv2
import numpy as np
import mediapipe as mp

# Set up paths
data_dir = "/content/drive/MyDrive/50471594_shreyadh_assignment02/data"
model_dir = "/content/drive/MyDrive/50471594_shreyadh_assignment02/model"
train_dir = os.path.join(data_dir, "train")
label_dir = os.path.join(data_dir, "label")
results_dir = "/content/drive/MyDrive/50471594_shreyadh_assignment02/results/train/"


# Initialize the hand detection model
with mp.solutions.hands.Hands(
        static_image_mode=True,
        max_num_hands=2,
        min_detection_confidence=0.7) as hands:

    # Loop through each image in the training directory
    for filename in os.listdir(train_dir):

        # Read the image from disk
        image = cv2.imread(os.path.join(train_dir, filename))

        # Convert the image to RGB
        image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

        # Process the image and detect hands
        results = hands.process(image)

        # Get the image height and width
        h, w, _ = image.shape

        # Loop through each hand detected in the image
        for hand_landmarks in results.multi_hand_landmarks:

            # Convert landmarks to pixel coordinates
            landmarks = [[int(l.x * w), int(l.y * h)] for l in hand_landmarks.landmark]

            # Extract the x and y coordinates of each finger
            fingers = {
                'thumb': [landmarks[2], landmarks[3], landmarks[4]],
                'index': [landmarks[5], landmarks[6], landmarks[7]],
                'middle': [landmarks[9], landmarks[10], landmarks[11]],
                'ring': [landmarks[13], landmarks[14], landmarks[15]],
                'pinky': [landmarks[17], landmarks[18], landmarks[19]]
            }

            # Loop through each finger
            for finger_points in fingers.values():

                # Extract the fingertip coordinates
                fingertip = finger_points[2]

                # Create a bounding box around the fingertip
                bbox_size = 200
                bbox_half = int(bbox_size / 2)
                bbox_tl = (fingertip[0] - bbox_half, fingertip[1] - bbox_half - 200)  # modified line
                bbox_br = (fingertip[0] + bbox_half, fingertip[1] + bbox_half - 200)  # modified line

                # Draw the bounding box on the image
                cv2.rectangle(image, bbox_tl, bbox_br, (0, 255, 0), 2)

        # Save the image with the bounding boxes
        cv2.imwrite(os.path.join(results_dir, filename), image)
```

Alternative models explored:

https://github.com/ultralytics/yolov3 - object detection

References:

https://developers.google.com/mediapipe/solutions/vision/hand_landmarker
https://opencv.org/

Task 4: Validation

Recall = 15/20 = 0.75
Avg IOU = 55%

Task 5: Testing

I have used the 5 images from the professor dataset, out of which I am getting it in 3 images perfect, which gives me the accuracy of about 60%, which gives in a total of evaluation metrics, as:
Recall ⅗ = 0.60
Average IOU = 44

Since, my colab was crashing on the 50 images of the professor dataset, so I tried on the 5 images on my detection algorithm, to give the above result.

Task 6: Future Scope

Fingerprint detection has been widely used for personal identification and verification purposes, such as for access control, law enforcement, and financial transactions. However, there are still several future scopes for fingerprint detection, including:

Improved accuracy: Although current fingerprint detection systems have a high level of accuracy, there is still room for improvement. Future developments may focus on increasing accuracy and reducing the false acceptance rate.

Real-time detection: Real-time fingerprint detection can be beneficial in situations where time is critical, such as in security applications. Future research may focus on developing real-time detection systems that can quickly identify and verify fingerprints.

Multimodal biometric systems: Multimodal biometric systems use multiple biometric factors, such as fingerprints, face recognition, and voice recognition, for personal identification and verification. Future research may focus on developing such systems that provide increased security and accuracy.

Non-contact fingerprint detection: Current fingerprint detection systems require physical contact with a sensor, which can be problematic in certain situations. Future

research may focus on developing non-contact fingerprint detection systems that can detect fingerprints from a distance, such as using infrared or optical sensors.