..

**Name :- Ayub Alam**

**Project Name :- Facial landmarks detection**

**Facial Landmark Detection Documentation**

The facial landmark detection script is a Python program that uses the dlib library to detect faces in an image and predict the positions of 68 facial landmarks. This documentation provides an overview of how to use the script and its functionalities.

**Installation:**

Ensure you have Python installed on your system.

Install the dlib library using pip:

pip install dlib

Install OpenCV for image processing:

pip install opencv-python

pip install numpy

**Usage:**

Download the pre-trained facial landmark predictor (shape\_predictor\_68\_face\_landmarks.dat) from the dlib website.

Modify the image\_path variable to specify the path to the image you want to process.

Run the script. The image with facial landmarks will be displayed.

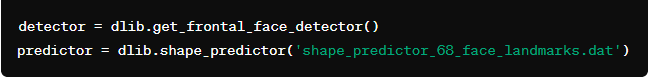
**Code :-**

This code uses the dlib library to detect faces in an image and then predicts and marks 68 facial landmarks on each detected face. Here's a step-by-step explanation:

**Step-1:** Import the necessary libraries: **dlib** for face detection and facial landmark prediction, **cv2** for image manipulation, and **numpy** for numerical operations.



**Step-2:** Load the pre-trained facial landmark detector from the **dlib** library. This detector is trained to detect faces in an image and predict 68 facial landmarks.



**Step-3:** Load the image using **cv2.imread()** function.



**Step-4:** Convert the image to grayscale, as the facial landmark detector expects a grayscale image.



**Step-5:** Detect faces in the grayscale image using the **detector** object.



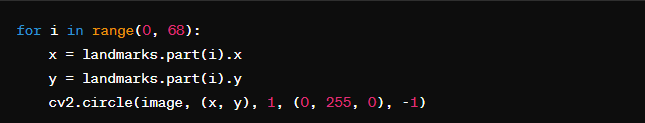
**Step-6:** Iterate over each detected face.



**Step-7:** Predict the facial landmarks for the current face using the **predictor** object.



**Step-8:** Iterate over each of the 68 facial landmarks and draw a circle at each landmark position on the original image using **cv2.circle()** function.



**Step-9:** Display the image with the facial landmarks using **cv2.imshow()** function.



**Step-10:** Wait for a key press to close the image window and then close all OpenCV windows using **cv2.waitKey(0)** and **cv2.destroyAllWindows()** respectively.



**Note:** This code demonstrates how to detect faces and predict 68 facial landmarks using the dlib library and mark them on the original image using OpenCV in Python.

**Result:-**

**Input\_image:- Output\_image:-**

**Limitations:**

The current implementation assumes that the input image contains only one face. Processing images with multiple faces may require additional logic.

The facial landmark positions may not be accurate for faces with extreme poses or occlusions.

**Conclusion:**

The facial landmark detection script provides a simple yet effective way to detect faces and predict facial landmarks in images. By following the provided instructions, users can quickly analyze facial features and integrate this functionality into their projects.