**Extract the outline of the face (without hair) removing all the background**

**Introduction**

Here we will see about the python function which will take an image as an input and detect the face present in that image and extract only the face without hair and background.

**Importing Important libraries**

As I am working with python programming language. It has beautiful inbuilt libraries to deal with images and computer vision problems.

**OpenCV**

A library of Python bindings designed to solve computer vision problems.

**dlib**

It's a landmark's facial detector with pre-trained models, the dlib is used to estimate the location of 81 coordinates (x, y) that map the facial points on a person's face. Basically it is a toolkit for making real world machine learning and data analysis applications such as face detection and facial landmark detection.

**scikit-image (skimage)**

It is used for Image Processing in Python. It is a collection of algorithms for image processing and computer vision. It provides utilities for converting between image data types.

**scipy.spatial.ConvexHull**

Hull means the exterior or the shape of the object. Therefore, the Convex Hull of a shape or a group of points is a tight fitting convex boundary around the points or the shape. Vertices : it will give the indices of points forming the vertices of the convex hull.

**Matplotlib**

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.

**Reading the image**

I have used the OpenCV, python library to read the image.

**Detecting the face present in the image**

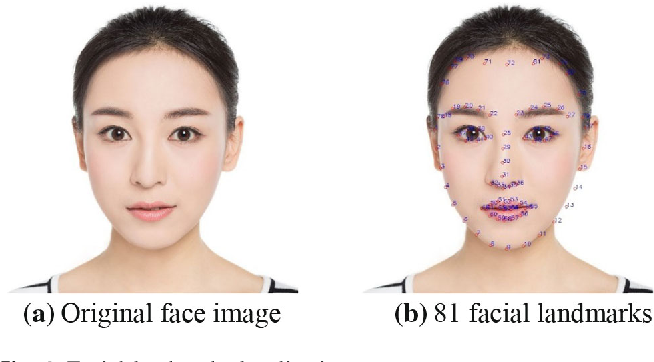
The library dlib comes with an inbuilt frontal face detector which detects faces present in the image. This detector is based on histogram of oriented gradients (HOG) and linear SVM. As we have almost all the faces facing front so I have used get\_frontal\_face\_detector() over cnn\_face\_detection\_model\_v1(). And also for the HOG based detector, don’t need to provide any file to initialize. For the CNN based detector needs to provide the weights file to initialize with.

Here while applying the detector I have used 1 which is the number of times it should upsample the image. By default, 1 works for most cases. (Upsampling the image helps to detect smaller faces).

Once the detection is done, we can loop over the detected face(s). To draw a box over the detected faces, we need to provide (x,y) — top left corner and (x+w, y+h) — bottom right corner to OpenCV. (0,255,0) represents the color of the box in BGR order (green in this case). 4 represents the thickness of the line.

**Shape prediction**

Here I have used pre pre-trained model to predict 81 landmarks on the faces such as eyes, eyebrows, nose, lips/mouth, jawline, forehead. Using this model we will get the (x,y) coordinates of all the required landmarks on the face. But these are in tuple format so we will convert them into a list of numpy arrays.



ConvexHull function will take the landmarks as input and give the indices of outline of the face. Using those outlines, indices skimage form a polygon by simply joining all those landmarks.

Then I have created one fully blank image. And using above indices of outline of face, simply cut down the original image in that region and paste in the same regione of the newly created blank image. So that we will have an image with only a face outline without hair and background.