Facial Landmarks Predictions

- Gaurav Sharma

ML based approach -

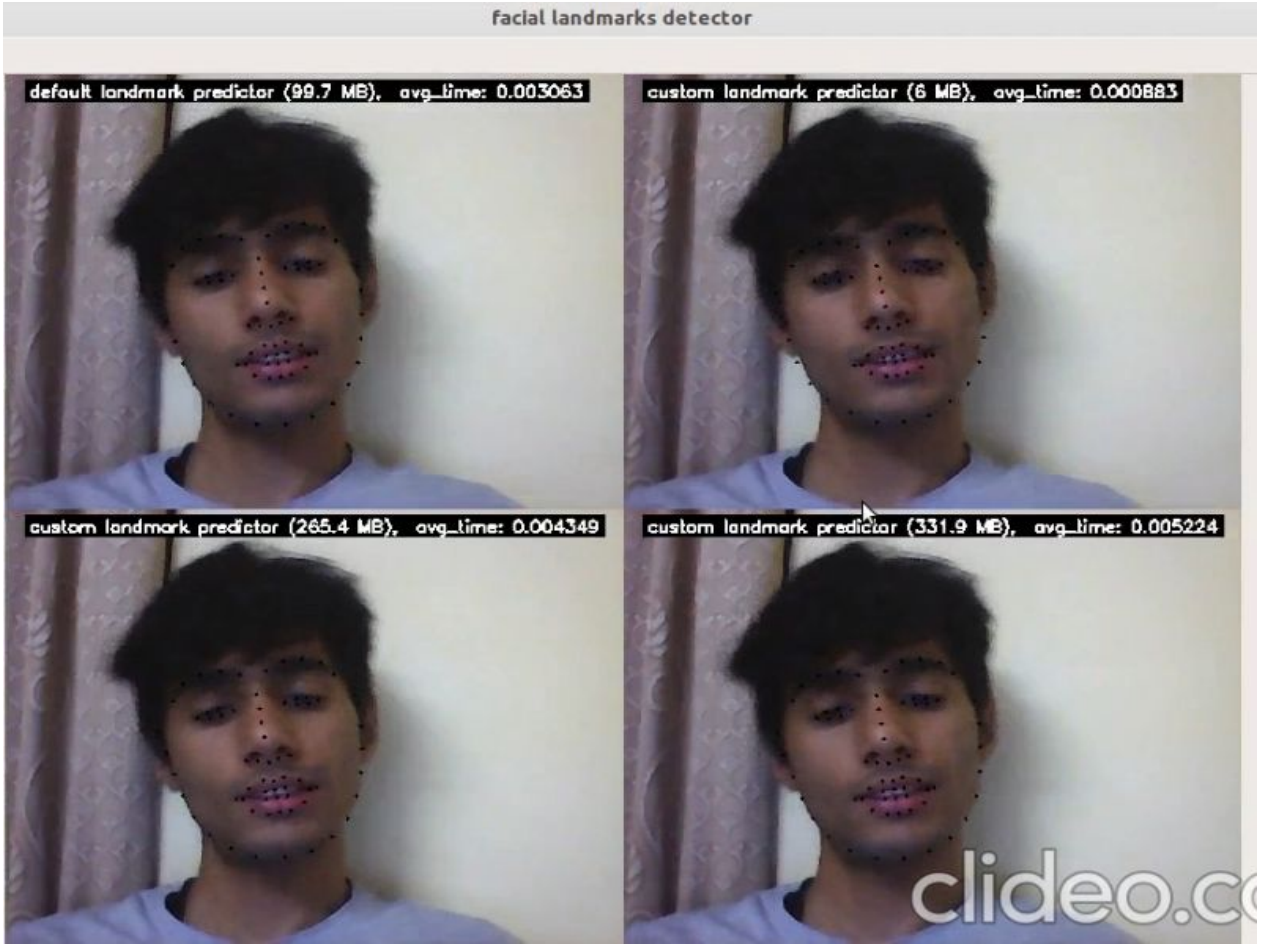
Before jumping straight to Deep learning I explored ML ways of approaching this problem and found that infact dlib is doing the same as well. Training a custom dlib facial landmarks predictor in dlib is much simpler than I thought. Dlib provides a very easy to use API for that. For this task, dlib utilizes the power of Ensemble models.

To estimate the landmarks location, the algorithm,

- Examines a sparse set of input pixel intensities (the features to the model)
- Passes the features into as **Ensemble of Regression Trees** (ERT)
- Refines the predicted locations to improve accuracy through a **cascade of regressors**

I trained 3 custom dlib facial landmarks predictors by changing various parameters like **cascade_depth**, **tree_depth**, **num_test_splits**, **oversampling_amount** etc. For this I used the **ibug_300w** dataset provided by dlib itself.

Here is a comparison of various custom and default dlib predictors,



Pros -

- Lighting fast
- Easy to train and test
- Works perfect for frontal faces.

Cons -

- Have only 68 landmarks, missing various important regions like the forehead which are important for many tasks (we can fix this by annotating new points and using more data).
- Doesn't perform well for left and right profile faces, also for faces with larger tilts (we can fix this by training more on non-frontal faces).

You can refer to this notebook for more details on training and evaluation of above custom models.
Refer this video to see the model comparison in more detail.
Refer this article for more details on dlib API for facial landmarks detection.

Deep-Learning based approach -

The task of predicting landmarks is very well achieved by deep-learning based approaches and outperforms traditional approaches in terms of performance. The simplest of them is to use any well known CNN model like **VGG**, **ResNet**, **DenseNet**, **InceptionNet** etc for regressing the landmarks.

Apart from this I found this MTCNN (Multi-task Cascaded CNN) which not only finds faces in an image and also 5 key landmarks on the face (2-2 for eyes and lips and 1 for nose tip). For this it uses a cascade of 3 CNNs, they call them 3 stages names as **P-Net**, **R-Net**, **O-Net**. It gives really good results in real time.

Refer this article for more details on MTCNN.

Pros -

- Takes more time as compared to dlib
- Deep learning approaches suits this kind of complex tasks

Cons -

- Currently supports only 5 facial landmarks (but we can extend this by annotating and using more data)