National Institute of Technology

Department of Computer Science and Engineering
Surathkal, Mangalore,
Karnataka - 575025

Face Recognition

June 16, 2017

Overview of Intelligent face recognition system using deep neural networks

Faces are detected irrespective of expressions and emotions. Detected faces are then fed into neural network to create dataset for clustering. After clustering every cluster has an unique labels, by which face recognition(validation) is performed.

Dependencies to be installed

1) Linux operating system (Ubuntu 16.04 LTS)
2) Python 2.7 (codes are tested with 2.7.13 comes by default with ubuntu)
3) Lua (recent stable version) and dependencies installation
    1. curl -R -O http://www.lua.org/ftp/lua-5.3.4.tar.gz
    2. tar zxf lua-5.3.4.tar.gz
    3. cd lua-5.3.4/
    4. make linux install
    5. luarocks install dpnn
    6. luarocks install nn
    7. luarocks install image

4) Installing torch

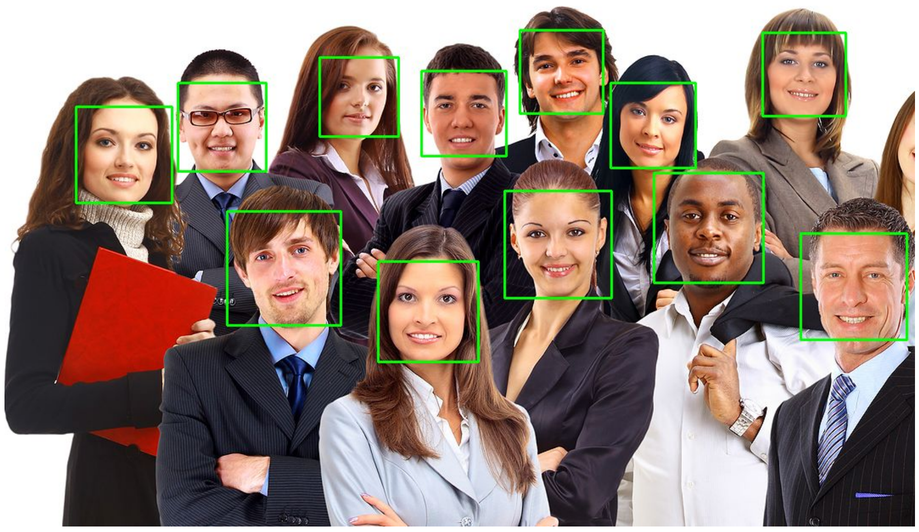
   1) In terminal, run commands WITHOUT sudo
   2) git clone https://github.com/torch/distro.git ~/torch --recursive
   3) cd ~/torch; bash install-deps;
   4) ./install.sh

1. Face detection

Two methods to detect the faces were enclosed. If pictures (.jpeg or .jpg format) are in directory of a file system in local machine then, to visualize how face detection is working, run face_detect.py faces(Testing purpose).

To run code : python face_detect.py

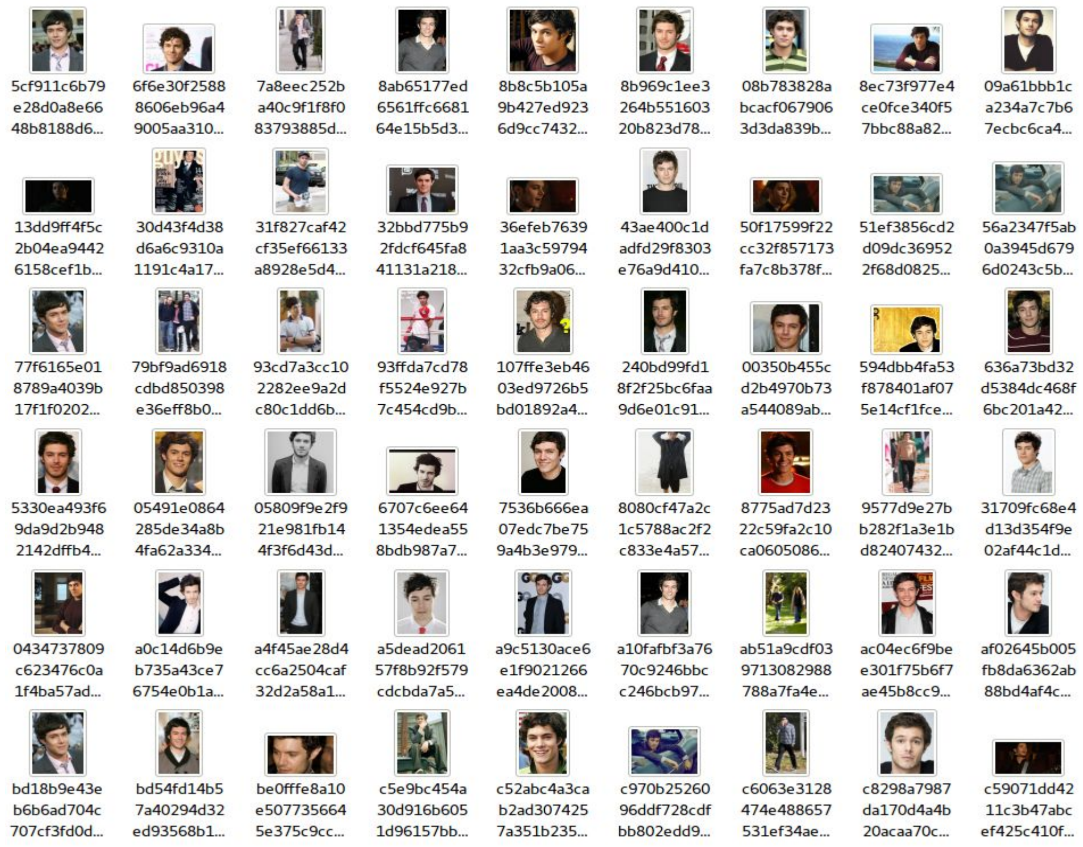
Sample output:



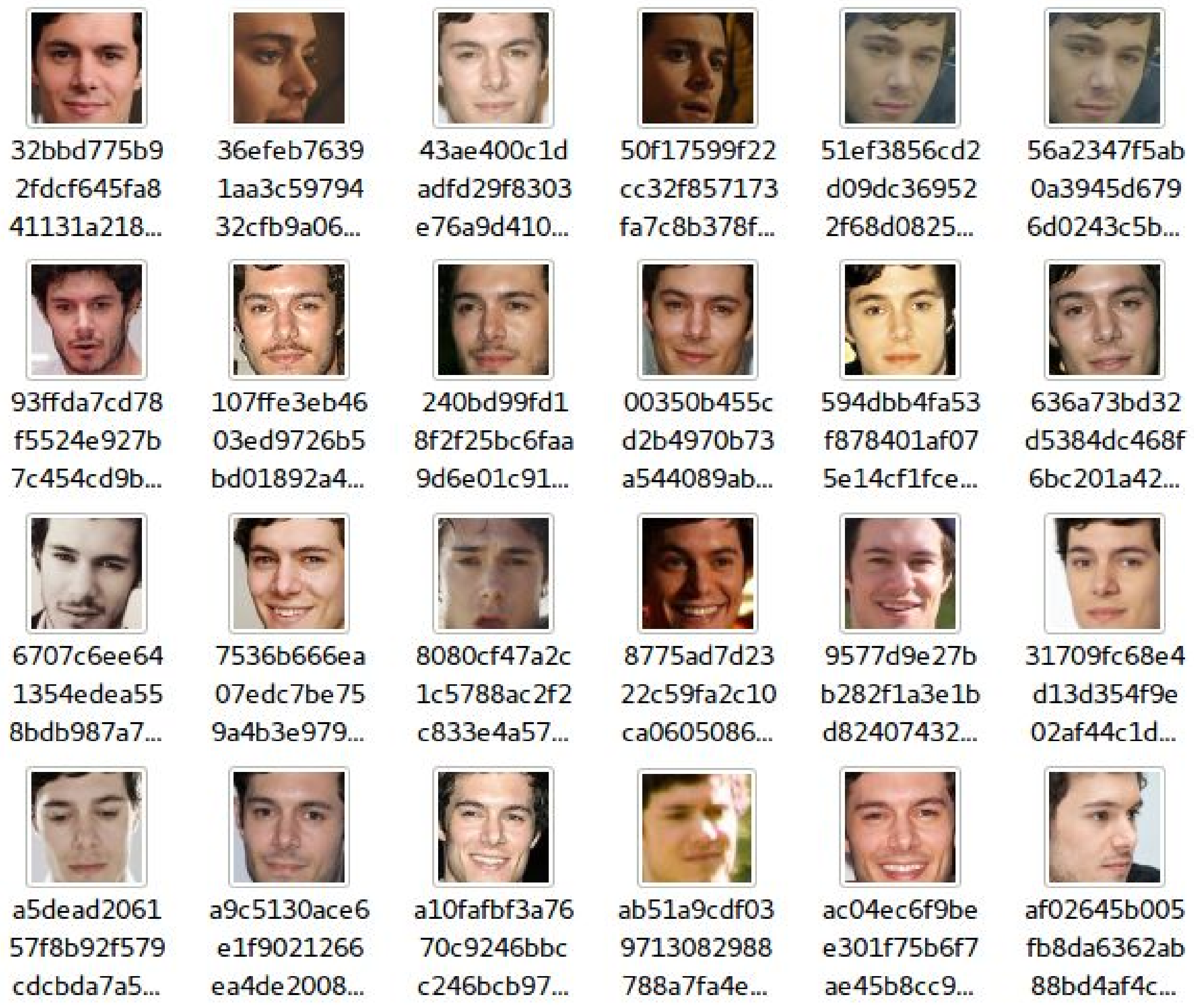
Dataset for face recognition is created by cropping facial images detected in parent picture(parent picture can also be group photo) and then feeded into neural network.Reverse mapping has to be done , when images are projected at phone (from cropped faces to its parent pictures - Database)

Method 1 : Run face_crop2.py to automatically detect faces and crop it to new directory in parent directory (Note: Haarcascade_frontalface_default.xml should be in parent directory while running face_crop2.py script).Haar cascade and dlib libraries are used to detect faces from the images(for more technical information refer code).

Sample : Parent directory containing single person photo or group photo


5cf911c6b79 e28d0a8e66 48b8188d6...
6f6e30f2588 8606eb96a4 9005aa310...
7a8eec252b a40c9f1f8f0 83793885d...
8ab65177ed 6561ffc6681 64e15b5d3...
8b8c5b105a 9b427ed923 6d9cc7432...
8b969c1ee3 264b551603 20b823d78...
08b783828a bcacf067906 3d3da839b...
8ec73f977e4 ce0fce340f5 7bbc88a82...
09a61bbb1c a234a7c7b6 7ecbc6ca4...
13dd9ff4f5c 2b04ea9442 6158cef1b...
30d43f4d38 d6a6c9310a 1191c4a17...
31f827caf42 cf35ef66133 a8928e5d4...
32bbd775b9 2fdcf645fa8 41131a218...
36efeb7639 1aa3c59794 32cfb9a06...
43ae400c1d adfd29f8303 e76a9d410...
50f17599f22 cc32f857173 fa7c8b378f...
51ef3856cd2 d09dc36952 2f68d0825...
56a2347f5ab 0a3945d679 6d0243c5b...
77f6165e01 8789a4039b 17f1f0202...
79bf9ad6918 cdbd850398 e36eff8b0...
93cd7a3cc10 2282ee9a2d c80c1dd6b...
93ffda7cd78 f5524e927b 7c454cd9b...
107ffe3eb46 03ed9726b5 bd01892a4...
240bd99fd1 8f2f25bc6faa 9d6e01c91...
00350b455c d2b4970b73 a544089ab...
594dbb4fa53 f878401af07 5e14cf1fce...
636a73bd32 d5384dc468f 6bc201a42...
5330ea493f6 9da9d2b948 2142dffb4...
05491e0864 285de34a8b 4fa62a334...
05809f9e2f9 21e981fb14 4f3f6d43d...
6707c6ee64 1354edea55 8bdb987a7...
7536b666ea 07edc7be75 9a4b3e979...
8080cf47a2c 1c5788ac2f2 c833e4a57...
8775ad7d23 22c59fa2c10 ca0605086...
9577d9e27b b282f1a3e1b d82407432...
31709fc68e4 d13d354f9e 02af44c1d...
0434737809 c623476c0a 1f4ba57ad...
a0c14d6b9e b735a43ce7 6754e0b1a...
a4f45ae28d4 cc6a2504caf 32d2a58a1...
a5dead2061 57f8b92f579 cdcbda7a5...
a9c5130ace6 e1f9021266 ea4de2008...
a10fafbf3a76 70c9246bbc c246bcb97...
ab51a9cdf03 9713082988 788a7fa4e...
ac04ec6f9be e301f75b6f7 ae45b8cc9...
af02645b005 fb8da6362ab 88bd4af4c...
bd18b9e43e b6b6ad704c 707cf3fd0d...
bd54fd14b5 7a40294d32 ed93568b1...
be0fffe8a10 e507735664 5e375c9cc...
c5e9bc454a 30d916b605 1d96157bb...
c52abc4a3ca b2ad307425 7a351b235...
c970b25260 96ddf728cdf bb802edd9...
c6063e3128 474e488657 531ef34ae...
c8298a7987 da170d4a4b 20acaa70c...
c59071dd42 11c3b47abc ef425c410f...

After cropping, new folder will be created in same directory with cropped faces, these faces should be transformed into vectors for clustering.


32bbd775b9 2fdcf645fa8 41131a218...
36efeb7639 1aa3c59794 32cfb9a06...
43ae400c1d adfd29f8303 e76a9d410...
50f17599f22 cc32f857173 fa7c8b378f...
51ef3856cd2 d09dc36952 2f68d0825...
56a2347f5ab 0a3945d679 6d0243c5b...
93ffda7cd78 f5524e927b 7c454cd9b...
107ffe3eb46 03ed9726b5 bd01892a4...
240bd99fd1 8f2f25bc6faa 9d6e01c91...
00350b455c d2b4970b73 a544089ab...
594dbb4fa53 f878401af07 5e14cf1fce...
636a73bd32 d5384dc468f 6bc201a42...
6707c6ee64 1354edea55 8bdb987a7...
7536b666ea 07edc7be75 9a4b3e979...
8080cf47a2c 1c5788ac2f2 c833e4a57...
8775ad7d23 22c59fa2c10 ca0605086...
9577d9e27b b282f1a3e1b d82407432...
31709fc68e4 d13d354f9e 02af44c1d...
a5dead2061 57f8b92f579 cdcbda7a5...
a9c5130ace6 e1f9021266 ea4de2008...
a10fafbf3a76 70c9246bbc c246bcb97...
ab51a9cdf03 9713082988 788a7fa4e...
ac04ec6f9be e301f75b6f7 ae45b8cc9...
af02645b005 fb8da6362ab 88bd4af4c...

Method 2 : Run script download.py to download from url (Note : create uri as per sample files given (file1.txt and file2.txt), then pass file name in list of download.py , all files will be downloaded in path mentioned), example: Run script download.py to get hollywood actresses and actors face pictures.

Creation and Cleaning dataset

Feed the cropped faces into neural network, which will provide output of 128 dimensional vector for each faces. These are considered as features of faces.

a) Run lua script as below (make sure that 'nn4.small2.v1.t7 ' file should be in directory where lua script is located),

```
th load_images.lua -d path_to_directory_containing_only_faces >> output_file.txt
```

This script will load images in buffer and resize according to specific size (96x96x3x1), this is default input size of images to be feeded into neural network.

As result, all face images in mentioned directory as parameter in above command will be converted into 128 vector value. Those data has to be filtered from default strings. (Note : This step was performed manually finally it should be automated, by opening every "output_file.txt" and removed unwanted strings refer file from_NN.txt (file which stores default output from NN) and refer "to_next_step.txt" (file which is modified) to check the modifications).

Convert "to_next_step.txt" file to csv file to load data frame in python for clustering algorithms. Run convert_csv.py as "python convert_csv.py" to convert "to_next_step.txt " file into "output.csv" file which will result in n x 128 array. Sample file is attached as output.csv

Semi supervised learning model: (Implementation logic)

Problem statement : Building unsupervised model for face recognition.

We have pre trained model of Convolutional neural network which is capable of recognizing faces with 90 to 95% accuracy in case of supervised classification. (Open face - Developed from paper FaceNet https://arxiv.org/pdf/1503.03832.pdf)

Initial setup includes clustering of labeled data , so that centroids and farthest point from each centroid (radial distance of each cluster) were stored as pairs in csv file.

Validation techniques :

Once faces are detected from parent image which are cropped into proper size(1,3,96,96) and fed to existing neural network and output is sliced from 25th layer (Convolutional neural network is of 26 layers and because last layer is fully connected which will not map to pre trained faces or nodes we are slicing data from 25th layer). The result of 1 face was 128 vector value which is compared with centroids and farthest point from csv to validate whether result belongs to any of existing clustered faces (Clustered in Initial setup).

Above case is trivial , so that this result is better accuracy. But if result does not belongs to any of the clusters then added to new list, This result is considered as centroid with the minimal radial distance as mean of radial distances of all existing centroids.

For next coming facial images to be recognized will be compared with this newly created centroid in above step. Algorithm is performed recursively.

Once data were clustered and enough data for single person is collected , we can train CNN again to create node in final layer (26th layer) of NN. This technique is called semi supervised learning.

Detailed methodology

Clustering 128 Dimensional data:

Clustering includes two phases, Training (clustering known faces) and validation. In clustering mechanism known faces are clustered (Training phase) for example , If there were 10 unique person faces cropped which are converted into feature vectors loaded as csv file (as many instance of unique person faces are required minimum 1:20 ratio) K means clustering algorithm is performed , which results in 10 clusters, it is mandatory to mention number of person at particular instance of time at training.

Validation : Once clusters are formed their centroid values are stored in csv file externally, where in other hand dataset for validation is passed to be compared with the clusters, here euclidean distance is taken between the validation data point to all other clusters(centroid and farthest point) , if data is nearer to the centroid of any cluster and less than the distance between farthest point from centroid of cluster , then data is added to that cluster. If not ,then it is added to new list of centroids with minimal radial distance as mean of radial distance of all other existing clusters. Further points comes for validation will be checked with newly formed centroid and its radial distance.

Pseudo code

TRAINING :

1. K Means Clustering (Dataset of K persons images) , K = Number of unique person.
2. Result : $<C_1,RD_1>$..........$<C_k,RD_k>$ stored in csv.

VALIDATION:

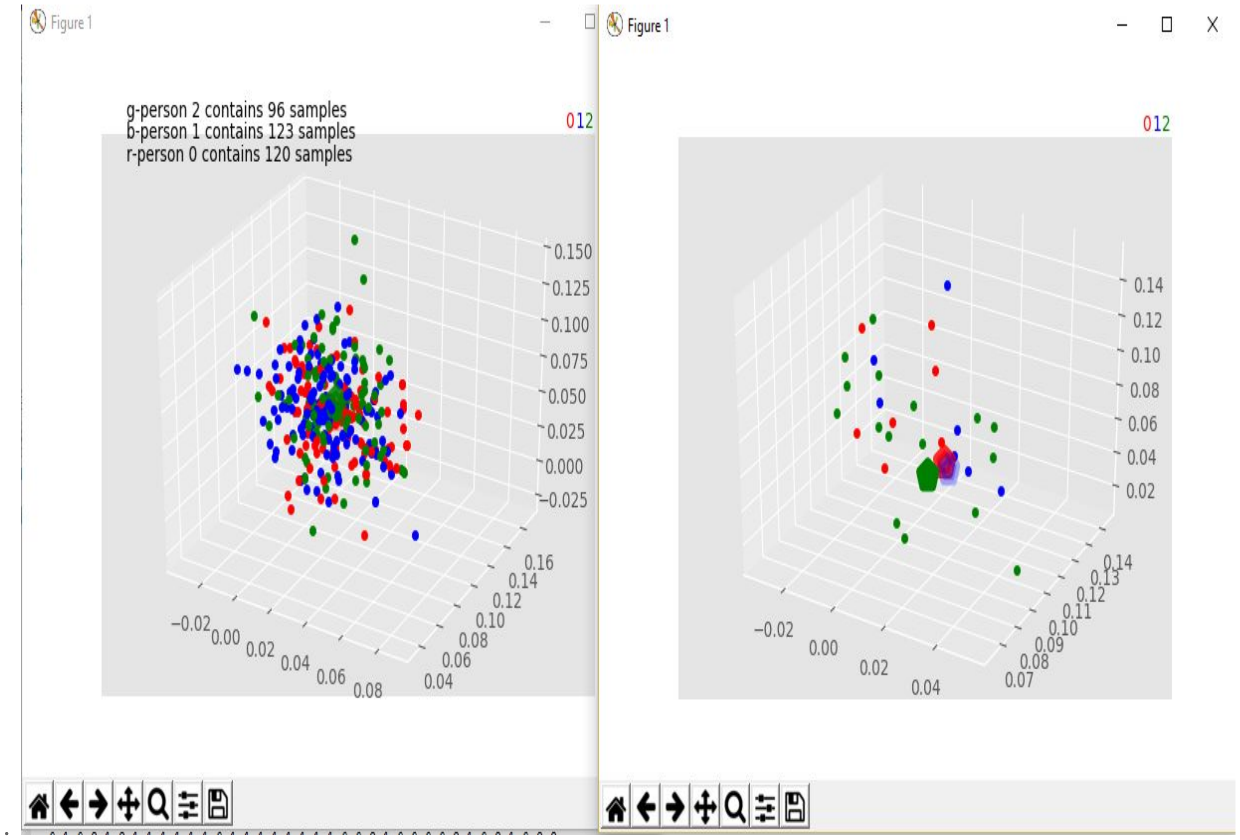
1. Start timer from 0 mins, each time validation starts
2. For each point in validation data set(V_1 to V_n), perform below :

```
IF (TIMER < 60 mins){
    FOR (i --> 1 to K) :
        Distance , Centroid = FUNCTION (MIN( Euclidean distance ($C_i$,$V_1$)) , $C_i$ )
    IF (RD(Centroid) > Distance)
        Add $V_1$ to C(Centroid)
    ELSE
        Add $V_1$ existing Result list as <$C_{new}$= $V_1$ , $RD_{new}$ = MEAN(Result list <$RD_i$>)
        K = K+1;
    }
    ELSE  TRAINING with updated K value.
```

Experiments performed

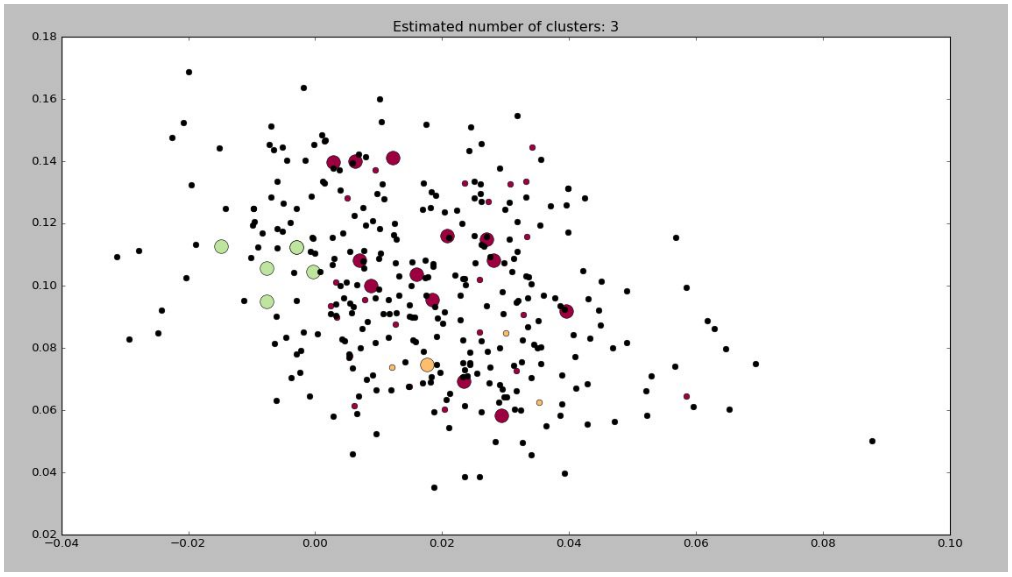
K means ++ : According to observation on k means algorithm , it will take local optimum value as its centroid so, results are inaccurate, So k means ++ clustering is opted which produces more effective results but not scalable for larger data set with accuracy of 60%.If number of unique person at particular instance is mentioned then accuracy can be increased.

Training and Validation results of k means ++ algorithm :



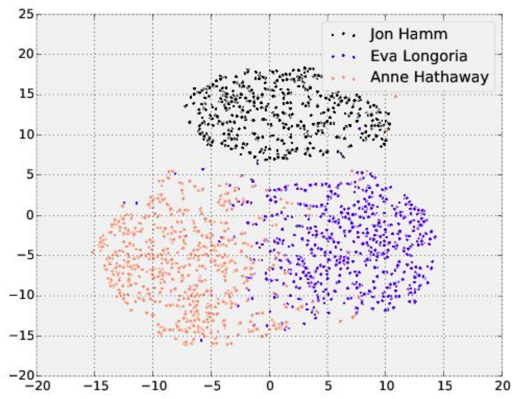
kNN algorithm : Implementation Kd tree and ball tree algorithm were giving more effective results than k means algorithm but for kNN classifier we need to mention number of classes at particular instance of time. So the problem again falls into supervised learning so we dropped this approach. Here classes unique person faces.

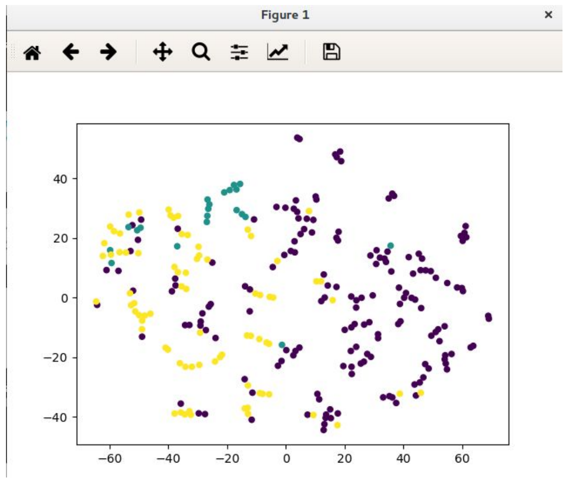
DbScan algorithm : Density based Clustering algorithm is not much efficient due to curse of dimensionality , which resulting in clusters of irrelavant sizes which cannot be processed further efficiently.



Validation Accuracy using t-SNE :

It is a non linear dimensionality reduction technique that is particularly well-suited for embedding high-dimensional data into a space of two or three dimensions, which can then be visualized in a scatter plot. Specifically, it models each high-dimensional object by a two- or three-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points.





Current work :

1. Experimentation with k means parallel algorithm for better scalability.

2. Local Binary Patterns Histograms for Face Recognition in unsupervised environment. http://bitsearch.blogspot.in/2013/02/unsupervised-face-clustering-with-opencv.html

Results :

Unsupervised model cannot be developed with 95% accuracy provided with very less data, time and computational power.

“ Google help thread on 2016 states - Google’s unsupervised model for 2000 images which was uploaded in photos app in one instance, takes 15 to 20 days to cluster similar faces using unsupervised technique, this feature called as “Group by faces” only available in specific countries.

Face Recognition in Unsupervised Environment is open research problem , Google has api for face detection only. Google Face recognition algorithm is not open sourced.

Google Brain team might be using “Variational Auto encoders” and “Generative Adversarial Neural Networks” to extract features of faces which requires exponentially large dataset of unlabeled faces and high computational power for better accuracies. These concepts are under research.

Papers published on 2017 under review , exactly suits our requirements

https://arxiv.org/pdf/1611.02648.pdf

https://arxiv.org/pdf/1606.05579.pdf