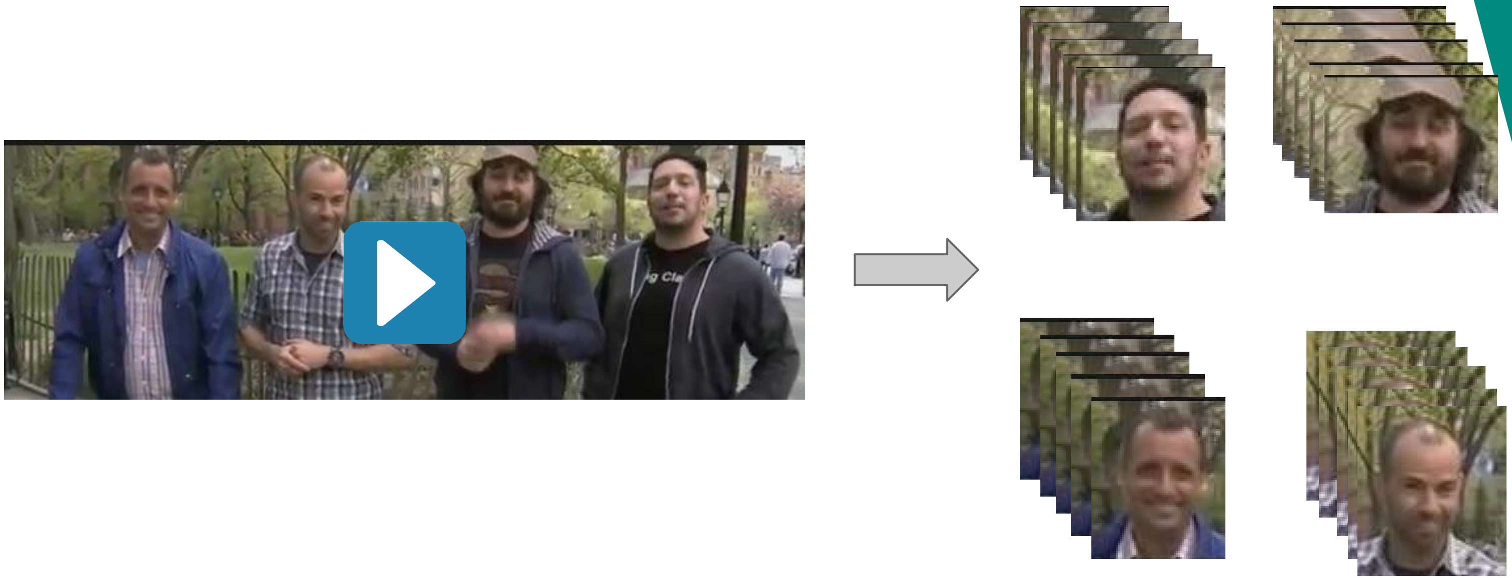
Unique face identification in a video

Problem Statement

Identify the number of unique characters in a movie clip by grouping similar faces together

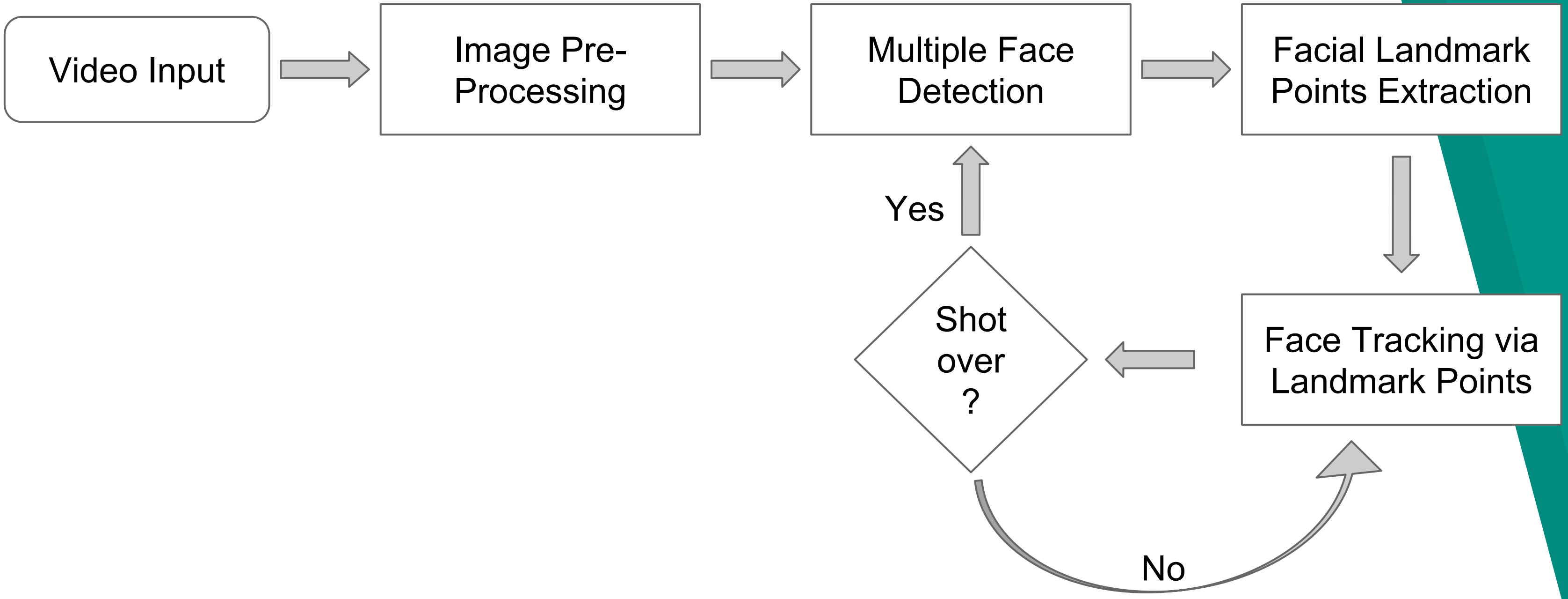
Problem Statement



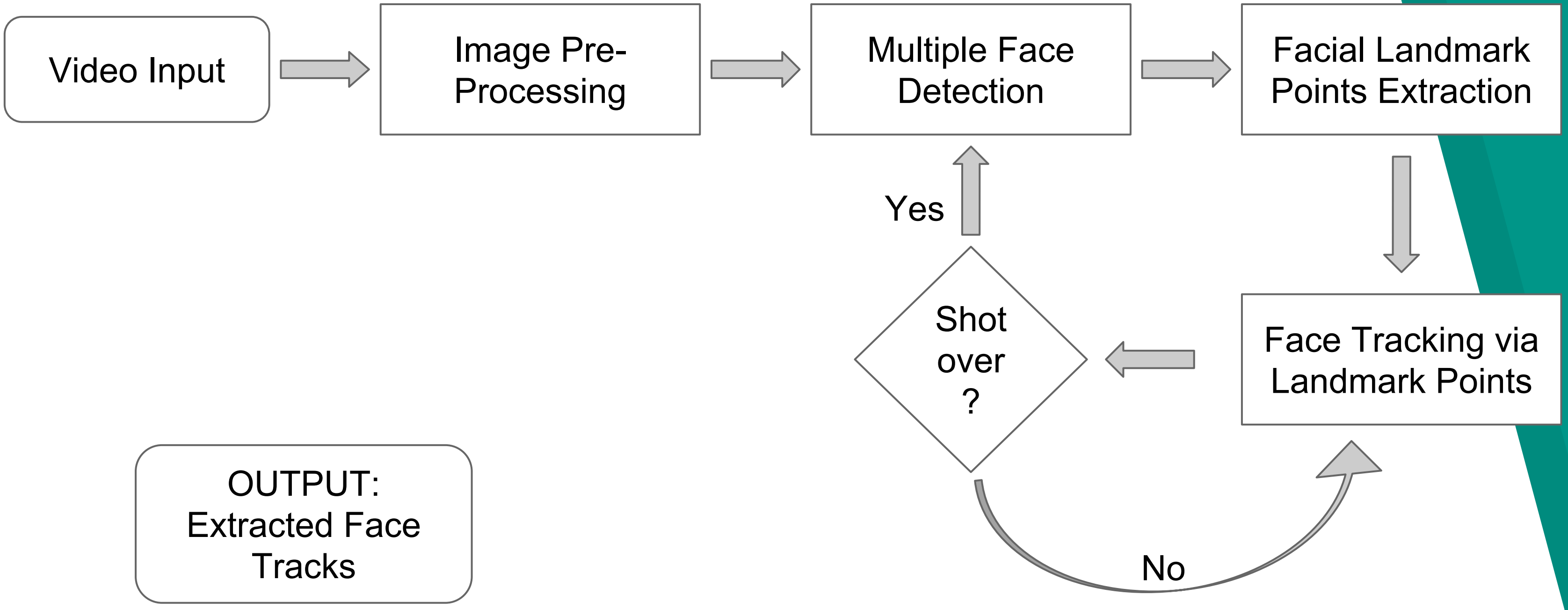
Relevance of the Problem

- Character Indexing
- Screen time estimation
- Video Analytics
- Surveillance

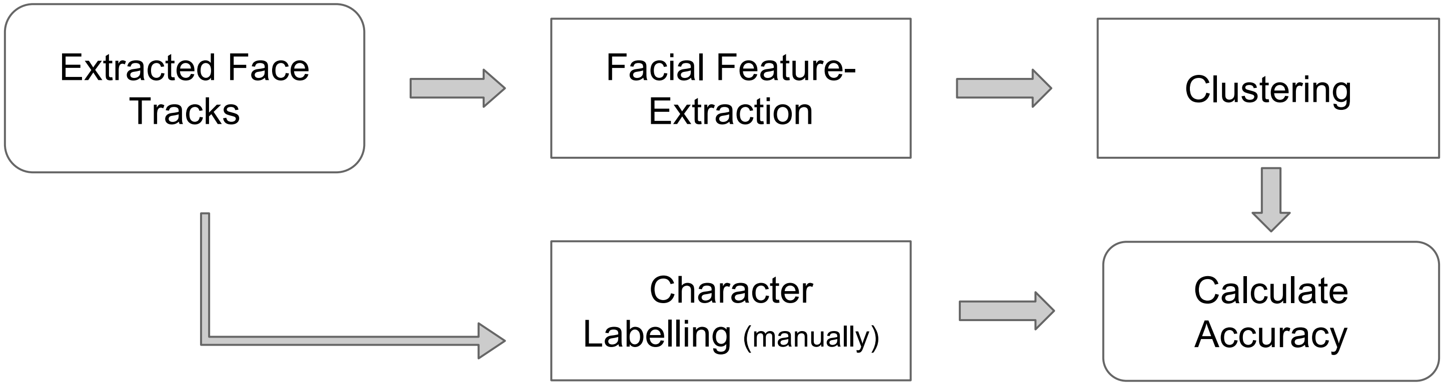
Block Diagram of Face Tracks Extraction:



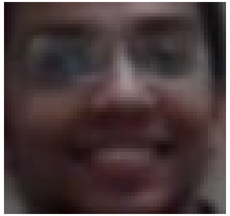
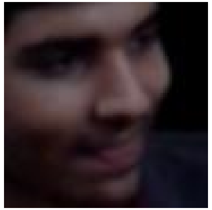
Block Diagram of Face Tracks Extraction:

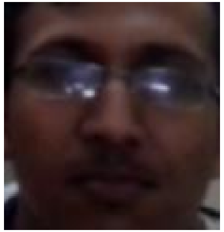


Block Diagram of Clustering the Face Tracks:



Video with four characters



Example

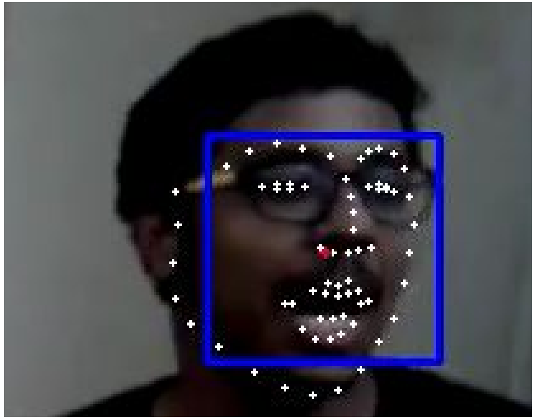
Video Stats and features:

- mp4, 720x480p, 30fps, 130 secs
- The video has 4 characters: 59, 62, 75, 81 tracks respectively
- Pose Variation
- Lighting Condition Variation
- Occlusion

Image Pre-Processing:

- Get a copy of frame
- Convert to Grayscale
- Gaussian Smoothing
- Histogram Equalization
- However

Face Detection and Landmark-Points Extraction:



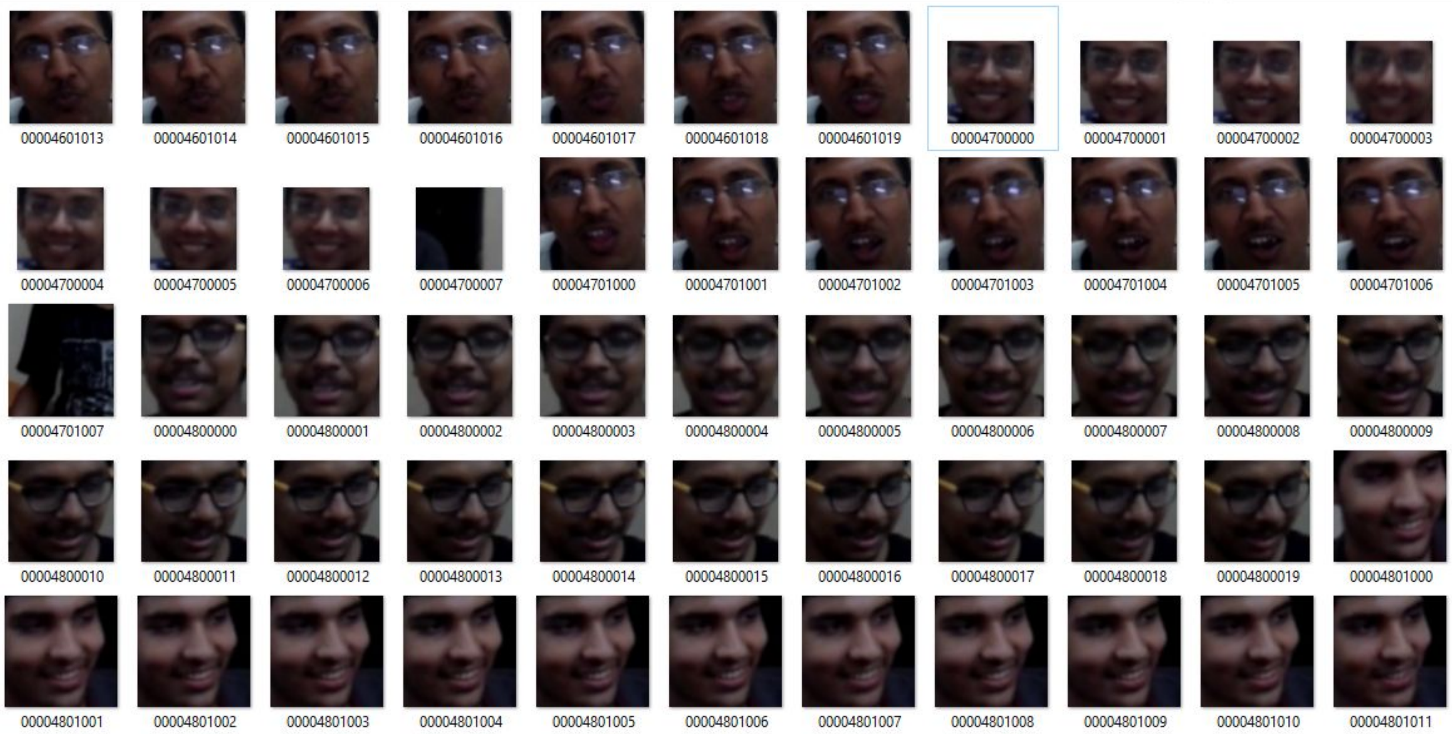
Face Detection and Landmark-Points Extraction:

- Viola Jones
- VJ replaced by Dlib frontal face detector
- Dlib landmark Point Detector

Face Tracking:

Track landmark points of each face using Lucas-Kanade Tracker, available in OpenCV

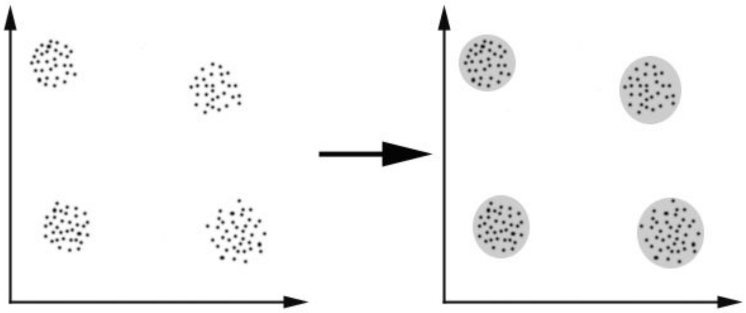
Extracted Faces:



Clustering:

- Unsupervised grouping of similar data

- Examples: K-Means Clustering, Spectral Clustering

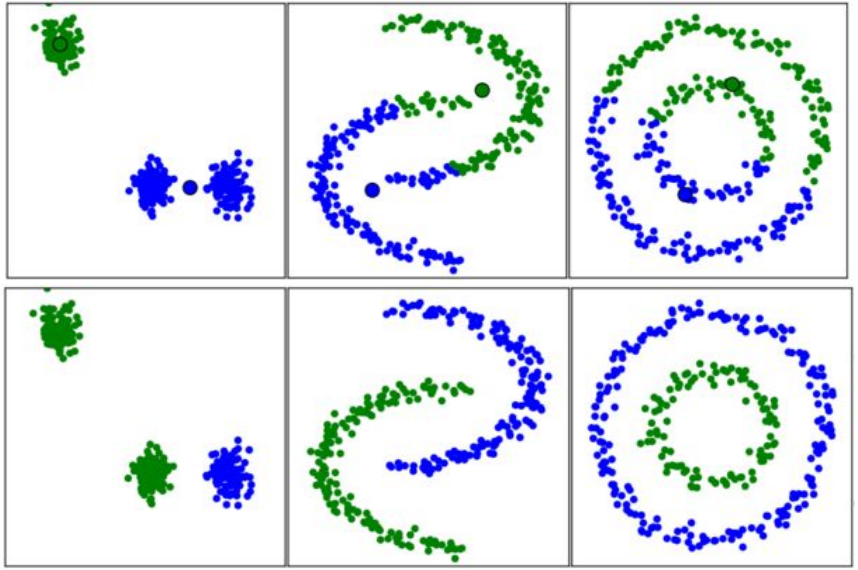


Reference- http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/
(A Tutorial on Clustering Algorithms)

Clustering Methods Used:

KMeans:

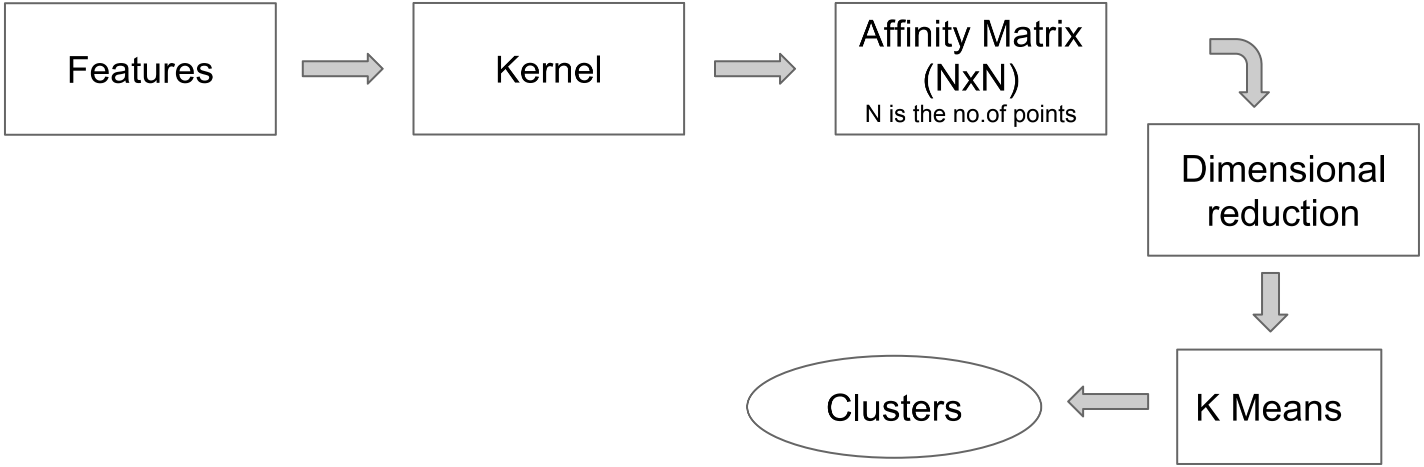
Spectral Clustering:



Reference-http://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/modules/clustering.html

Spectral Clustering- SC:

- Performance better than K-Means in general
- However, performs poorly for highly non-uniform distribution of classes



Affinity Matrix:

- Stores ‘similarity’ between every pair of data points

- ‘Similarity’ is calculated using kernels:

  a. Linear Kernel: $K(x_i, x_j) = x_i . x_j^T$

  b. Intersection Kernel: $K(x_i, x_j) = \Sigma \min(x_i, x_j)$

  c. Laplacian Kernel: $K(x_i, x_j) = \exp(-\gamma|x_i - x_j'|)$

  d. RBF Kernel: $K(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2)$

What is a Kernel?

An Example -

Feature Vector1	100	65	90
Feature Vector 2	77	37	200
Feature Vector 3	20	156	109
Feature Vector 4	233	100	40

Intersection Kernel

$$\Sigma \min(x_i, x_j)$$

Feature 1	100	65	90
Feature 2	77	37	200

$$77 + 37 + 90 = 204$$

What is a Kernel?

An Example -

Feature Vector1	100	65	90
Feature Vector 2	77	37	200
Feature Vector 3	20	156	109
Feature Vector 4	233	100	40

Affinity Matrix

$\Sigma \min(x_i, x_j)$

Features ⇨ ⇩

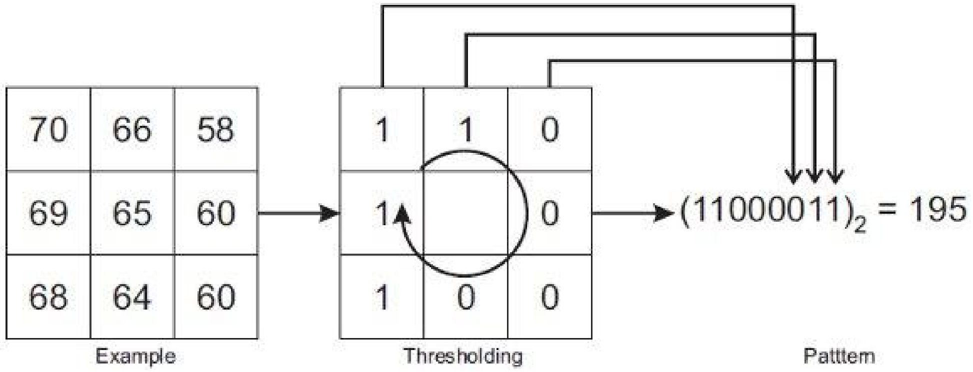
	1	2	3	4
1	255	204	175	205
2	204	314	166	154
3	175	166	285	160
4	205	154	160	373

Very basic Features Used as input:

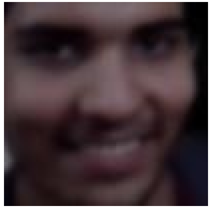
- Scaled raw RGB image, averaged over face track
- LBP(Local Binary Pattern) image of the scaled averaged image
- Histogram of pixel intensities of Raw Image
- Histogram of pixel intensities of LBP Image

Local Binary Pattern- LBP:

- Texture feature
- Illumination Invariant



Accuracies on Basic Features:



- RGB Raw (102x102 x3): SC, int. Kernel: 59.56%
- Histogram (128 bins x3): SC, linear: 57.04%

- Raw LBP Image (100x100 x3): SC, int. Kernel: **66.78%**
- Histogram (128 bins x3): SC, linear: 59.92%

We set 66.78% as our benchmark.

Advanced Features Used as input:

- Uniform LBP-TOP
- HOG-TOP
- Principal Component Analysis (PCA) on above mentioned features for dimension reduction.

Uniform Local Binary Pattern:

- Give unique pattern-value to only LBP values with less than or equal to two 01/10 bit-transitions.
  Eg: 00010000 (2 transitions)

- 58 such LBP patterns are possible, so 58 pattern values From 1 to 58.

- All other LBP patterns go into a pattern value 0.
  Eg: 01001000 (4 transitions)

- Total 58+1=59 patterns possible for Uniform LBP.

Uniform Local Binary Pattern:

0 0 0 1 0 0 0 0

2 transitions

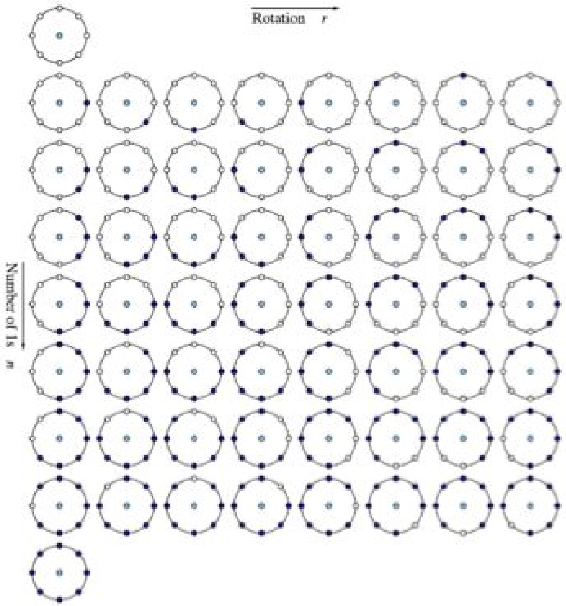
0 1 0 0 1 0 0 0

4 transitions

1 1 1 1 0 0 0 0

1 transition

Uniform Local Binary Pattern:



Reference:http://quantgreeks.com/uniform-local-binary-pattern-in-matlab/

Histogram of Oriented Gradients (HOG):

- For every pixel, get x and y derivatives

-1/3	0	1/3
-1/3	0	1/3
-1/3	0	1/3

x derivative mask

140	130	145
160	156	144
150	140	152

part of Image

-1/3	-1/3	-1/3
0	0	0
1/3	1/3	1/3

y derivative mask

Here, f_x=-6.33, f_y=9

Histogram of Oriented Gradients (HOG):

- Then calculate the magnitude and direction of the gradient
- Distribute it into the 8 radial bins

$f_x = -6.33$, $f_y = 9$

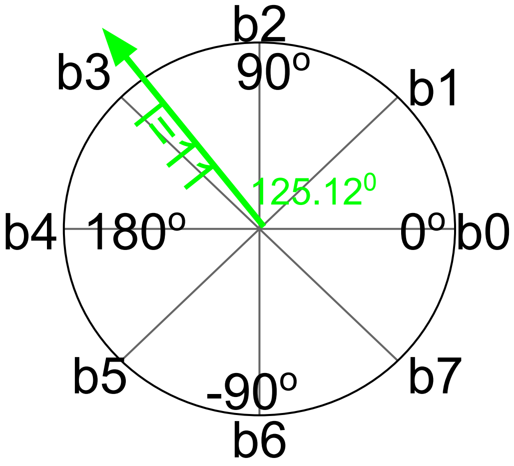
Mag = 11.00

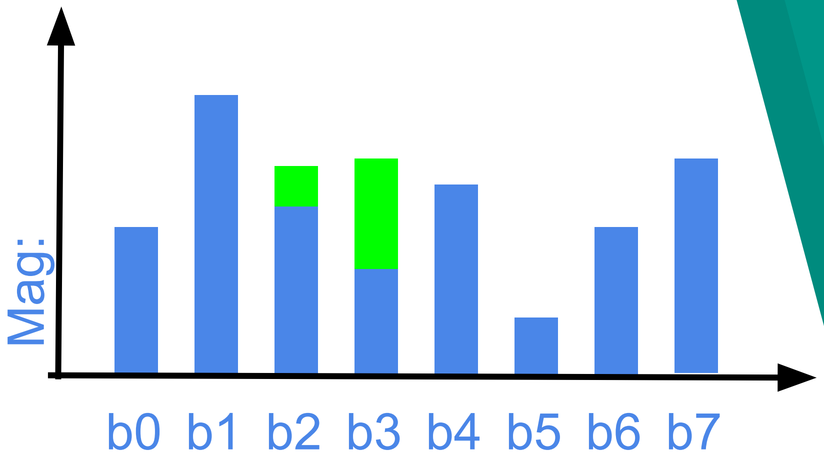
Angle = 125.12°

Histogram of Oriented Gradients (HOG):

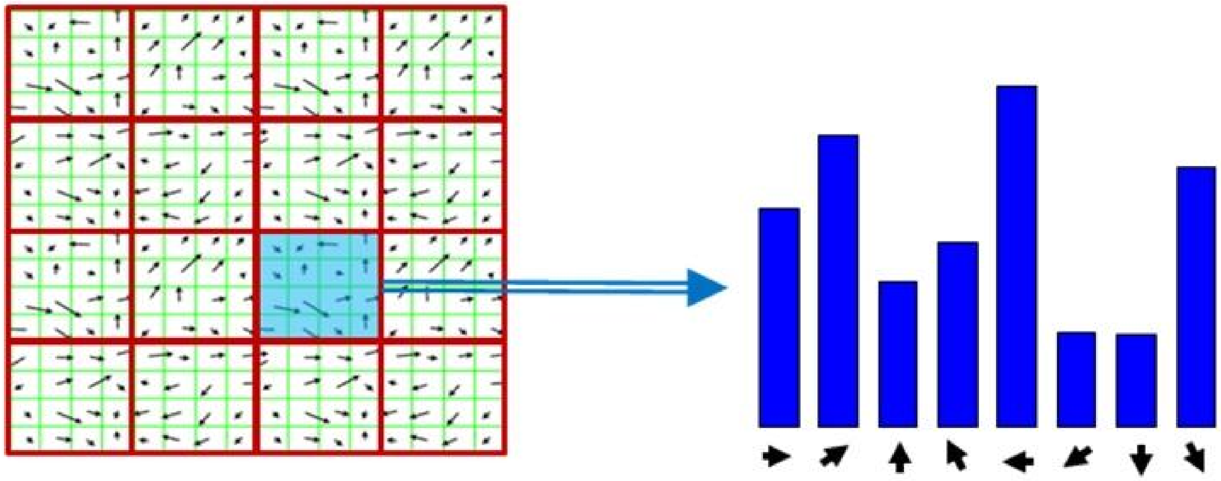
Bin 135°(b3): add 11 x |90-125.12| ÷ 45 = 8.58

Bin 90°(b2): add 11 x |135-125.12| ÷ 45 = 2.41





Histogram of Oriented Gradients (HOG):



https://gilscvblog.com/2013/08/18/a-short-introduction-to-descriptors/

Uniform LBP/HOG
TOP (Three Orthogonal Planes):

- TOP stands for Three Orthogonal Planes: XY, XT, YT
- For each sectional block, obtain normalized histogram of LBP/HOG for each color, for all three planes.

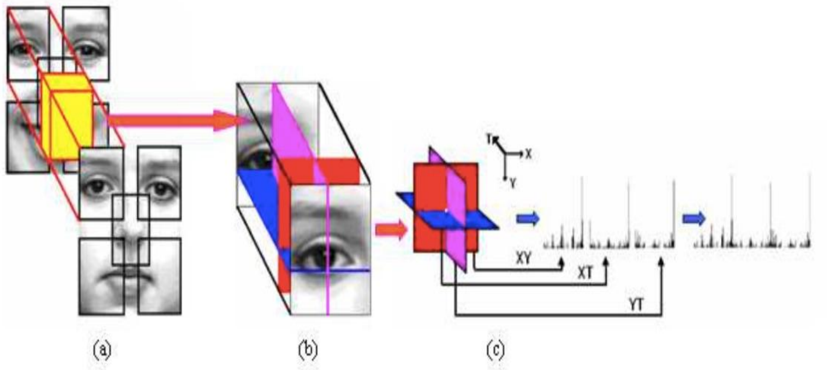


Fig. 9. Features in each block volume. (a) Block volumes; (b) LBP features from three orthogonal planes; (c) Concatenated features for one block volume with the appearance and motion

Reference- http://www.ee.oulu.fi/research/mvmp/mvg/files/pdf/pdf_740.pdf
(Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions)

LBP/HOG-TOP
Histogram Concatenation:

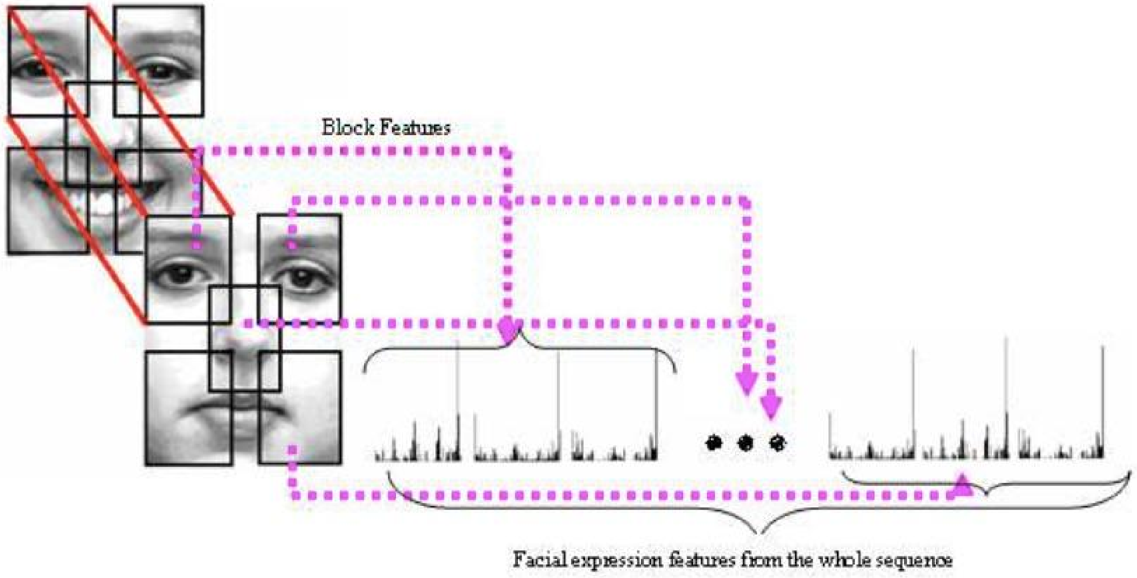


Fig. 10. Facial expression representation.

Reference- http://www.ee.oulu.fi/research/mvmp/mvg/files/pdf/pdf_740.pdf
(Dynamic Texture Recognition Using Local Binary Patterns with an Application to Facial Expressions)

Accuracies on Advanced Features:

LBP-TOP (59x9x3):

- 3x3, SC, int. Kernel: 69.67%
- After 10% overlap: 3x3, SC, int. Kernel: 67.87%

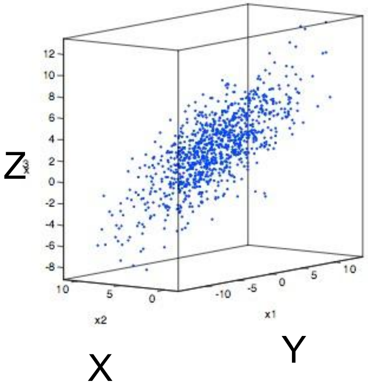
HOG-TOP (8x9x3):

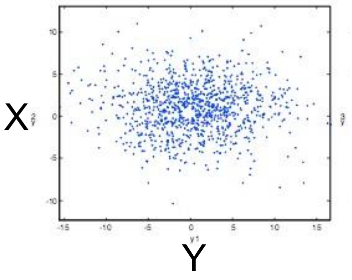
- 3x3, SC, int. Kernel: **73.64%**
- After 10% overlap: 3x3, SC, int. Kernel: 67.14%

Principal Component Analysis (PCA):

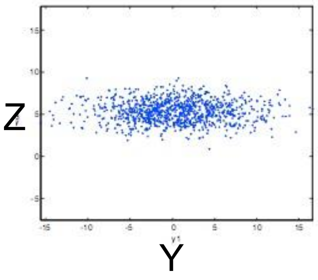
Correlated variables → Linearly Uncorrelated variables (Principal Components)

Dimensional Reduction

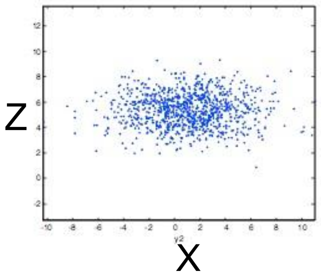




Maximum Variance



Minimum Variance



Accuracies after applying PCA:

LBP-TOP:

- n=11, γ = 0.0001, rbf kernel: 79.06%
- With 10% overlap, n=11, γ = 0.0001, rbf: **82.31%**

HOG-TOP:

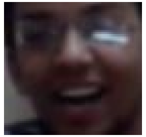
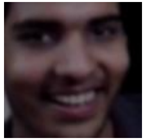
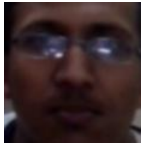
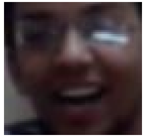
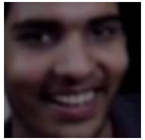
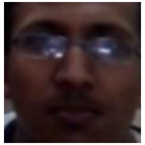
- n=10, γ = 0.1, rbf kernel: 71.84%
- With 10% overlap: n=12, γ = 0.1, rbf: 66.78%

Benchmark was 66.78 at feature size 30000!

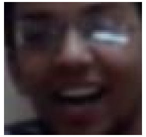
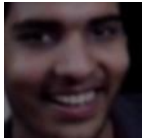
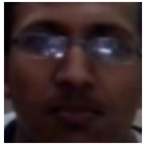
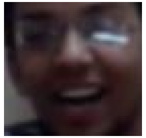
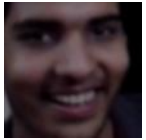
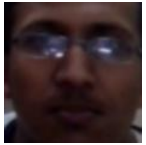
Confusion Matrix for Best Case:

Predicted class →

Actual class ↓

				
	44	1	8	6
	0	52	10	0
	0	1	74	0
	0	0	23	58

Confusion Matrix for Best Case:

				
	74.6%	2%	13.5%	10.7%
	0	83.9%	16.1%	0
	0	1.3%	98.7%	0
	0	0	28.%4	71.6%

Possible Improvements:

- ❑ Our ultimate goal is to achieve Real-time Online Clustering, which is still to be achieved. Current work acts as a base which can easily be used for further research in this area.
- ❑ Using a better detector (eg: CLM detector)
- ❑ Improving the tracking algorithm.
- ❑ Real-time shot-detection.

Possible Improvements:

- ❏ Using different features (eg: SIFT extracted features) for Clustering.
- ❏ Using supervised learning on large face datasets to obtain better features.
- ❏ Utilising GPU programming for faster execution.

References:

- Shot Detect software from http://johmathe.name/shotdetect.html
- *Zhao G & Pietikäinen M (2007)* Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(6):915-928.
- Dlib library- frontal face detector, landmark face detector
- Python clustering http://scikit-learn.org/stable/modules/clustering.html
- PCA in Python: http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html
- An Efficient Approach for Clustering Face Images: Charles Otto, B.K.Noblis, Anil Jain: https://www.nii.ac.jp/pi/n7/7_53.pdf
- Koji YAMAMOTO1, Osamu YAMAGUCHI2, and Hisashi AOKI3: Fast face clustering based on shot similarity for browsing video 1,2,3: https://www.nii.ac.jp/pi/n7/7_53.pdf
  Corporate Research and Development Center, Toshiba Corporation

References:

- Viola Jones Object Detection Framework: https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework
- Wikipedia, for getting basic idea of things: https://en.wikipedia.org/wiki/Main_Page
- Paul Viola, Michael J Jones: Robust Real-Time Face Detection: http://www.vision.caltech.edu/html-files/EE148-2005-Spring/pprs/viola04ijcv.pdf
- Mubarak Shah Video Lecture, Optical Flow: https://www.youtube.com/watch?v=5VyLAH8BhF8&index=6&list=PLmyoWnoyCKo8epWKGHAm4m_SyzoYhslk5
- David G. Lowe: Distinctive Image Features from Scale-Invariant Keypoints: https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf
- Mubarak Shah video lecture, HOG: https://www.youtube.com/watch?v=0Zib1YEE4LU

Thanks

Submitted by:
Amit Gupta
Lakshay Garg
Prakhar Kulshreshtha