Face Clustering in Videos : GMM-based Hierarchical Clustering using Spatio-temporal Data

Subhradeep Kayal
Department of Information and Computer Science
Aalto University
Espoo, Finland
subhradeep.kayal@aalto.fi

***Abstract*—In recent years, an increase in multimedia data generation and efficient forms of storage have given rise to needs like quick browsing, efficient summarization and techniques for information retrieval. Face Clustering, together with other technologies such as speech recognition, can effectively solve these problems. Applications such as video indexing, major cast detection and video summarization greatly benefit from the development of accurate face clustering algorithms. Since videos represent a temporally ordered collection of faces, it is only natural to use the knowledge of the temporal ordering of these faces, in conjunction with the spatial features extracted from them, to obtain optimal clusterings. This paper is aimed at developing a novel clustering algorithm, by modifying the highly successful hierarchical agglomerative clustering (HAC) process, so that it includes an effective initialization mechanism, via an initial temporal clustering and Gaussian Mixture Model based cluster splitting, and introduces a temporal aspect during cluster combination, in addition to the spatial distances. Experiments show that it significantly outperforms HAC while being equally flexible.**

Keywords—Face Clustering, Gaussian Mixture Model, Hierarchical Clustering

I. INTRODUCTION

The problem of face recognition from still images has greatly evolved since the early solutions [1] [2]. Some of the successful methods have been reviewed quite well in [3]. Face recognition algorithms have been applied to surveillance and biometrics with ever-increasing efficiency. Apart from the aforementioned areas of application, human faces are extremely important in the context of multimedia information, such as videos, and are considered as high-level semantic features. With the advent of the information age and the explosion in the availability of such multimedia data, the clustering of faces in videos has become a parallel problem with recognition. In clustering, the interest lies in discovering the set of different classes of faces. It has many real-world applications including video indexing [4], extraction of principal characters or major cast detection from a video [5], building a face database [6] and semi-supervised learning [7]. Thus, face clustering from videos has generated a lot of research interest in the recent years.

In this paper, the clustering experiments are done on the images detected from a Finnish news broadcast clip. The traditional algorithm of hierarchical agglomerative clustering (HAC) [8] is compared with its novel variant. This new algorithm makes suitable use of the temporal information available in the video. It starts by creating clusters solely based on temporal similarity, allows a Gaussian Mixture Model (GMM) to check the integrity of each such cluster, and divide it into smaller clusters if necessary, and concludes by performing hierarchical agglomerative clustering on the clusters verified/created by the Gaussian Mixture Model. In the last step, a novel form of HAC is performed, where the temporal distance matrix is taken into account as well, alongwith the spatial distance matrix. The results of clustering are analyzed using the methodology stated in [9], wherein it can be observed that the proposed method improves the results of the HAC greatly.

II. PROCESS OVERVIEW

The overall process overview is shown in the following figure. Faces are detected from frames, which are extracted from the video at the rate of 1 per second, using a Viola-Jones face detector [10]. These detections are filtered, using a simple and fast PCA-eigenspace based method [2], to remove any non-face areas. Simple texture features are then extracted from each face and used in the clustering process.

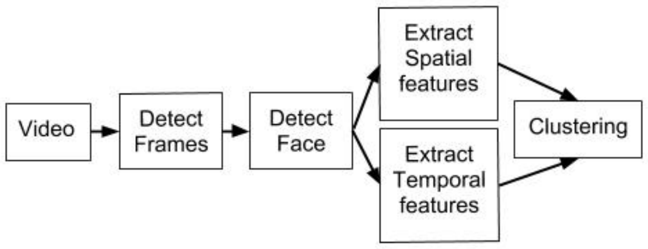

Fig. 1. General Process Overview

III. FACE DETECTION AND EXTRACTION OF SIMPLE FEATURES

A. Filtering of non-face regions

Once the frames have been captured from the video, at the rate of 1 frame per second or 1 frame per 25 frames (assuming the frame rate is 25 fps), the Viola-Jones face detector [10] is run on them to extract all the face-like regions from the frame. These regions may contain some non-face regions as well. In order to remove them, a simple PCA-based filtering method was implemented [2]. A subset of the extracted images, containing only face regions, was chosen and projected to the 'face-space' using PCA. Then, a region was accepted as a face if the distance between its projection on the 'face-space' and itself was less than a threshold.

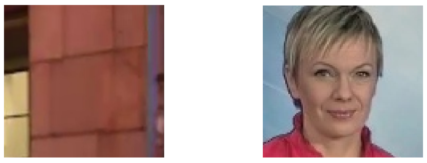

Fig. 2. Non-face and face regions (left-right)

B. Feature extraction: Gabor Features

It is important to extract features which are simple enough to be easily understood and quickly calculated, as well as powerful enough to describe and discriminate the face image from which they have been extracted. One such type of simple descriptors are the Gabor features, which have been used widely in face recognition since the ground-breaking work in [11].

The Gabor filter is a linear filter which mimics the human visual system in its ability to assess spatial orientation and signal frequency. The Gabor feature vector is the absolute value of the result of convolving each image with a Gabor filter at a particular scale and frequency.

In the spatial domain, the 2D Gabor filter is formed by the modulation of a Gaussian kernel by a sinusoidal plane wave [12]. Amongst the many representations of the Gabor filter, the most common in context of face recognition is:

$$\phi_{f,\theta,\gamma,\eta}(x,y)=\frac{f^2}{\pi\gamma\eta}e^{-(\alpha^2x'^2+\beta^2y'^2)}e^{j2\pi f x'} \qquad (1)$$

where,

$$x'=x\cos\theta+y\sin\theta \qquad (2)$$
$$y'=-x\sin\theta+y\cos\theta \qquad (3)$$

Here, f is the central frequency of the sinusoidal wave, θ is the anti-clockwise rotation of the Gaussian and the plane wave, and α , β determine the sharpness of the Gaussian. $\gamma=\frac{f}{\alpha}$ and $\eta=\frac{f}{\beta}$ keep the ratio between frequency and sharpness a constant.

Instead of a single filter at a specific scale and orientation, a bank of filters is usually used. A filter-bank effectively captures most of the available information in an image. For face recognition, past studies suggest the use of a total of 40 filters in the filter bank, with 5 scales and 8 orientations:

$$f_u=\frac{f_{max}}{\sqrt{2^u}},\theta=\frac{v}{8}\pi,u=0,..,4,v=0,..,7 \qquad (4)$$

Each 2D output of convolution, that of an image with a component filter, is concatenated row-wise to produce 40 different row vectors, which are then concatenated to form one single high-dimensional feature vector for each image.

In this paper, the result of convolution per filter is resized to 15x10 pixels, such that the feature vector per face image is 15x10x40=6000 dimensional.

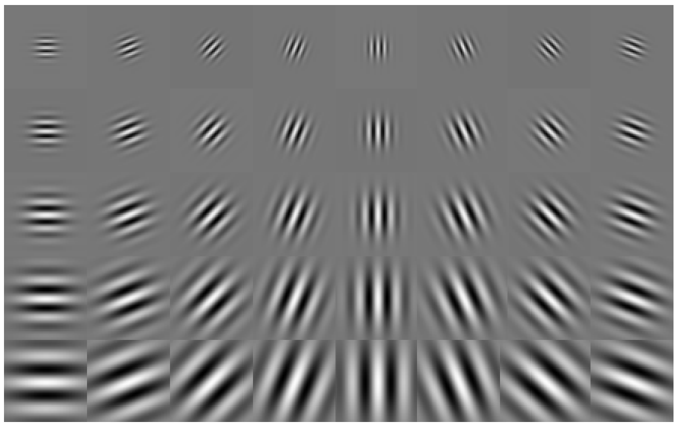

Fig. 3. Absolute values of the Gabor filters at various scales and orientations

IV. Discussing the Preliminaries

This section explains the modified spatio-temporal HAC algorithm. The component methods are explained first, followed by the explanation of the main algorithm to implement the novel HAC.

A. Gaussian Mixture Models (GMM) and the EM-algorithm

In an attempt to make the paper self-contained, an outline of the expectation-maximization (EM) algorithm is provided for learning the parameters of a Gaussian Mixture Model (GMM).

Gaussian Mixture Models (GMM) are mixtures of two or more Gaussians, which are linearly superposed. If suitably tuned, GMMs can model any data distribution to an arbitrary degree of accuracy. A K -component GMM, i.e., a GMM with K component Gaussians, is given by:

$$p(x)=\sum_{k=1}^{K}\pi_k N(x|\mu_k,\Sigma_k) \qquad (5)$$

where the constituent Gaussians have their own mean and covariance, mixed with a degree π_k known as the mixing coefficient, such that,

$$\sum_{k=1}^{K}\pi_k=1 \quad , \quad 0\leq\pi_k\leq 1 \qquad (6)$$

Before performing the Expectation-Maximization (EM) algorithm to fit the GMM to the training data, it is important to define a few more variables. A K -dimensional binary random variable z is defined, having a 1-of- K representation, in which a particular element is 1 and all others are 0. Then,

$$p(z_k=1)=\pi_k \qquad (7)$$

Since z uses a 1-of- K representation, it can be written as,

$$p(z)=\prod_{k=1}^{K}\pi_k^{z_k} \tag{8}$$

Similarly,

$$p(x|z)=\prod_{k=1}^{K}N(x|\mu_k,\Sigma_k)^{z_k} \tag{9}$$

Having defined eq. 8 and 9, another quantity, known as 'responsibility', is now defined, which is conditional probability of z given x . Using the Bayes' theorem:

$$p(z_k=1|x)=\gamma(z_k)=\frac{p(z_k=1)p(x|z_k=1)}{\sum_{j=1}^{K}p(z_j=1)p(x|z_j=1)} \tag{10}$$

Putting the value of eq. 8 and 9 in eq. 10,

$$\gamma(z_k)=\frac{\pi_k N(x|\mu_k,\Sigma_k)}{\sum_{j=1}^{K}\pi_j N(x|\mu_j,\Sigma_j)} \tag{11}$$

The above equation can also be written to represent the value of 'responsibility' for each data point x_n :

$$\gamma(z_{nk})=\frac{\pi_k N(x_n|\mu_k,\Sigma_k)}{\sum_{j=1}^{K}\pi_j N(x_n|\mu_j,\Sigma_j)} \tag{12}$$

Now, it is desired to find the values of $\boldsymbol{\pi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, to 'fit' the GMM to the training data, thereby making it a problem of parametric density estimation. An easy way to do it is to maximize the log-likelihood of these parameters, given the data, and this is what the EM algorithm does. The log-likelihood is given by:

$$L(\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}|\boldsymbol{X})=\sum_{n=1}^{N}\ln\{\sum_{k=1}^{K}\pi_k N(x_n|\mu_k,\Sigma_k)\} \tag{13}$$

where, K is the number of mixture components and N is the number of data points.

Now, we have defined all the necessary equations to provide an overview of the EM algorithm. The derivation of the EM algorithm is straightforward and proceeds by differentiating eq. 13 with respect to $\boldsymbol{\pi}$, $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, setting the results to zero, and solving the resultant equations. It can be found out by referring to [13]. The outline of the algorithm is as follows:

1. Initialize values of mixing coefficients, means and covariances.
2. **E-step**: Evaluate responsibilities using eq. 12 and the current parameter values.
3. **M-step**: Re-estimate the parameter values using eq. 14-16 :

$$\mu_k^{new}=\frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})x_n \tag{14}$$

$$\Sigma_k^{new}=\frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(x_n-\mu_k^{new})(x_n-\mu_k^{new})^T \tag{15}$$

$$\pi_k^{new}=\frac{N_k}{N} \tag{16}$$

where, N_k is the number of points belonging to the k^{th} component.

4. Evaluate log-likelihood using eq. 13 and check for convergence. If not converged, return to step 2.

After convergence of the EM algorithm, each data point is assigned to a cluster from 1 to K , depending on the probability of the point belonging to that particular component Gaussian.

Apart from parameter estimation, another problem with using GMMs, due to their parametric nature, is the selection of the number of components. This is a crucial step in the context of this paper, since GMMs are used, in the final algorithm, to split the clusters initially made by simple temporal ordering (for details check section V A), which form the initial clusters for HAC. In this paper, the true number of components are selected based on the minimization of the Bayesian Information Criterion (BIC) [14].

Formally, BIC is defined as,

$$BIC=-2\ln L+k\ln(n) \tag{17}$$

where, L is the maximized value of the log-likelihood (see eq. 13), obtained from the EM algorithm, k is the number of parameters to be estimated, and n is the sample size.

In this paper, GMMs are fitted to the data using the EM algorithm, for components varying from $K=1$ to M , and the value of K for which the value of BIC, from eq. 17, is minimum, is chosen to be the true value. Thus, for $K=1$, the data cluster which the GMM operates upon is not split at all, and for $K=C$,such that $1\leq C\leq M$, the cluster is split into C disjoint clusters.

B. Hierarchical Agglomerative Clustering (HAC)

Hierarchical clustering [8] simply tries to create a hierarchy of clusters. It is the method of partitioning the data points by constructing a nested set of partitions represented by a cluster tree. The agglomerative principle is a bottom-up approach which works by combining smaller clusters to form larger ones. Originally, HAC starts by placing each point in it's own cluster and combines clusters according to some similarity criterion. The flexibility in selecting the similarity criterion and the distance metric is a great strength of HAC.

HAC works following the given procedure:

1. Calculate the distance matrix d_s for the given data points using the spatial features, such that $d_s(i,j)$ gives the distance between points i and j . Commonly used distance metrics are euclidean, cityblock, cosine, etc.

2. Initialize each data point as a singleton cluster C_i .
3. Combine the nearest pair of clusters, C_i and C_j , according to some similarity measure, looking at the distance matrix calculated in step 1. Some common similarity measures are single-link (SL), complete-link (CL) and average-link (AL). Eq. 18-20 describe these aforementioned similarities:

$$SL = min\{d(a,b) : a \in C_i, b \in C_j\} \tag{18}$$

$$CL = max\{d(a,b) : a \in C_i, b \in C_j\} \tag{19}$$

$$AL = \frac{1}{|C_i||C_j|} \sum_{a \in C_i} \sum_{b \in C_j} d(a,b) \tag{20}$$

4. Continue merging until all points belong to a single final cluster.

The required number of clusters is selected by partitioning the cluster tree at the appropriate level. A pictorial example of how HAC works is given below,

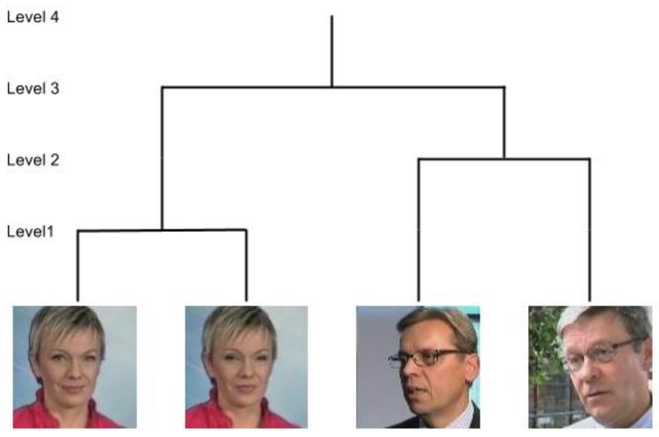

Fig. 4. HAC with four data points (faces)

V. Proposed Algorithm (GMM-THAC)

Having defined how clustering works via GMM and HAC, the proposed algorithm is now defined. Since the algorithm works via an initialization process based on an initial temporal ordering, followed by a EM-GMM based cluster re-estimation, and integrates temporal information in the hierarchical clustering procedure, it is termed as 'GMM-THAC'.

A. Initial Temporal Clustering

Once the faces have been detected and filtered to remove possible non-face areas, an initial temporal clustering is performed based on the time-stamp of the detected faces (in this paper, the frame number) by a simple windowing approach. A window of size w is defined, which is the maximum allowable difference between two consecutive face time-stamps for them to belong to the same temporal cluster. The value of w is dependent on the rate at which the frames are grabbed in the face detection process. For a rate of grabbing 1 frame per second, a good value of w is 5; when the rate increases to 5 frames per second, w may be set to 2.

Formally,

1. Set a value for w .
2. Sort the face time-stamps in ascending order.
3. Set the value for cluster index, $C=1$.
4. Put the face-sample corresponding to the first time-stamp in the sorted order in cluster $C=1$.
5. Check the difference between consecutive time-stamps. If difference is less than w , assign the face-sample a cluster label C . Otherwise, increase C by 1 and assign it as the label, i.e., put the sample in a new cluster.

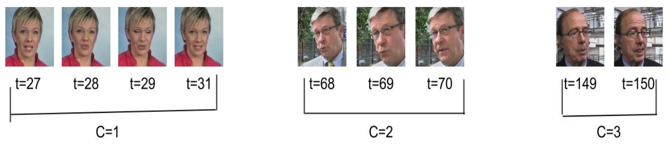

Fig. 5. Initial temporal clustering illustrated

B. EM-GMM reclustering

After the temporal clustering step, each cluster is verified by the BIC score of a K -component GMM with fitted to it (for details check section IV A). The value of K is varied from 1 to M , assuming that each temporal cluster may contain the face-samples of at most M different people. The value of M is chosen depending upon the rate of frame grabbing and the window size w . For a larger window size, the temporal clusters may be more 'mixed-up', and hence a larger value of M is chosen. In this paper, a good value of M is found to be 5.

Then,

1. For each temporal cluster,
   i. Successively fit GMMs to the data points in the cluster, varying the number of components from 1 to M .
   ii. Choose the correct number of components as K_c , corresponding to the value of the lowest BIC score.
2. Recluster the data by fitting a GMM with K_c components to it, i.e., split the cluster into K_c disjoint clusters.

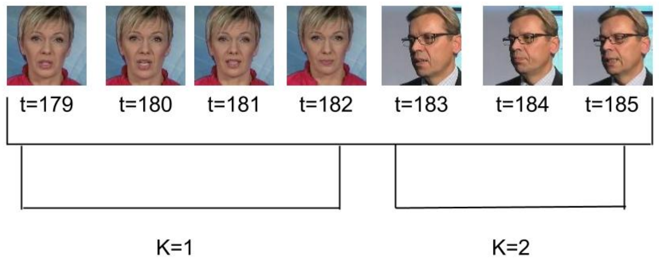

Fig. 6. Cluster splitting by EM-GMM

C. Hierarchical Agglomerative Clustering with Spatio-Temporal Distances

As explained previously (in step 1 of section IV B), traditional HAC uses the spatial distance matrix, calculated from the features extracted from the data points, during the process of clustering. In this paper, the HAC algorithm is modified to include the temporal distances between samples as well.

It may be noted that with an increase in the number of iterations while combining the clusters, the effect of the temporal distances should ideally decrease and the spatial distances should be more dominant for clustering. Otherwise two very similar samples (,i.e., very low spatial distance) occurring with a large time-difference between them (,i.e., very high temporal difference), say at the beginning and the end of the video (for example, the face image of the news reader), might not be combined. Thus, the temporal distance matrix is weighted with a factor which exponentially decreases.

The THAC works in the following way:

1. Calculate the distance matrix d_s for the given data points using the spatial features, as in step 1 of the original HAC algorithm. In this paper, cosine distances are used.

2. Calculate the temporal distance matrix d_t for the data point time-stamps, where

$$d_t(i,j)=|t_i-t_j| \tag{21}$$

   where, i, j are data points (face-samples), and t_i, t_j are the time-stamps (frame number labels) of those data points where, *iter* is the iteration number and N_{sample} is the sample size of the entire dataset..

3. The temporal distance matrix is weighted with an exponential decay term:

$$d_{tnew}=d_t\times\exp\left(-\frac{iter}{N_{samples}}\right) \tag{22}$$

   where, *iter* is the iteration number and N_{sample} is the sample size of the entire dataset.

4. Initialize each point to it's own singleton cluster C_i.

5. Find the nearest pair of clusters, according to some similarity measure operating on both the spatial and the temporal distance matrices, and combine them. Different similarity measures can be independently used for the spatial and the temporal data matrices, and the results may be combined.

   For example, in order to use the average-link criterion on both, the average-link similarities must be calculated (using eq. 20) for both the spatial and the temporal distance matrices. Then, the nearest pair of clusters are those for which the combined average-link similarity is minimum.

6. Continue merging until all the points belong to a single cluster.

The resultant clusters formed by the EM-GMM operation on the initial temporal clusters, act as the seed clusters for the THAC algorithm. The final number of clusters is selected to be 8, for comparison with the ground-truth.

VI. Dataset Description

The dataset [15] comprises 217 face-like images detected from the frames grabbed at the rate of 1 per second from a Finnish TV news broadcast clip. Of the 217 images, 211 are face images and 6 are wrongly detected non-face images, which get filtered out (see section III A). The 211 images form 8 disjoint clusters describing 8 different people (ground truth).

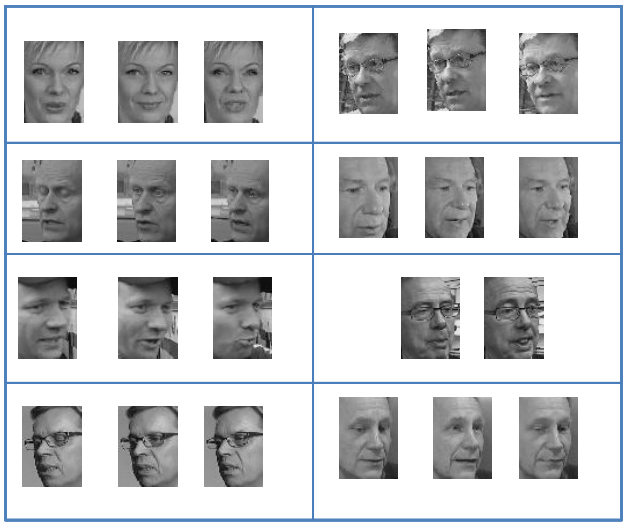

Fig. 7. Representative samples from the dataset, grayscaled. [15]

VII. Mechanism of Cluster Analysis

There exist many ways to evaluate clustering quality: intrinsic methods, which measure the within and between cluster proximities, and extrinsic methods, which use manual ground-truth classification. In this paper, the work in [9] is followed in order to measure the quality of clustering for a particular algorithm.

An evaluation measure based on the purity and inverse-purity of clusters is proposed in [9]. The clustering created by the various algorithms in question is the 'estimated clustering', whereas that made manually is the 'ground-truth clustering'. Then, 'estimated purity' (EP) and 'ground-truth purity' (GTP) are defined as:

$$EP=\frac{1}{D}\sum_k max_j|E_k\cap GT_j| \tag{23}$$

$$GTP=\frac{1}{D}\sum_k max_j|GT_k\cap E_j| \tag{24}$$

where, $D=N_{sample}$ is the number of detections, and GT_i, E_i are the i^{th} ground-truth and estimated clusters respectively.

Thus, a higher EP means less chance of a cluster getting 'mixed', or including faces of different people. Similarly, a higher GTP means less likelihood of a person's face appearing in multiple clusters. However, neither alone is a good criterion for judgement. Thus, an F-measure is defined. But before that, the EP and GTP measures should be scaled so that they are between 0 and 1.

Considering the case of a single cluster with all the faces, we take a look at eq. 23 and 24. It can be understood that GTP=1 and,

$$EP_{min} = \frac{max_j |GT_j|}{D} \tag{25}$$

Similarly, for the case where each image is a singleton cluster, EP=1 and,

$$GTP_{min} = \frac{|GT|}{D} \tag{26}$$

Using eq. 25 and 26 to scale the values of EP and GTP:

$$EP_n = \frac{EP - EP_{min}}{1 - EP_{min}} \tag{27}$$

$$GTP_n = \frac{GTP - GTP_{min}}{1 - GTP_{min}} \tag{28}$$

Now, the F-measure is constructed as a final determinant of algorithm performance. The F-measure is given as:

$$F = 2\frac{EP_n \times GTP_n}{EP_n + GTP_n} \tag{29}$$

Finally, eq. 29 is used to calculate the F-score of the HAC, EM-GMM and GMM-THAC algorithms.

VIII. RESULTS

In this paper, algorithms tested for clustering performance on the aforementioned dataset are:

1. Gaussian Mixture Model based clustering (EM-GMM).
2. Hierarchical Agglomerative Clustering with a cosine distance metric and average-link similarity measure (HAC).
3. The novel variant of HAC, without taking into account the temporal distance matrix (GMM-HAC).
4. The novel variant of HAC proposed in the paper, operating on both the spatial and the temporal distance matrices, with cosine similarity and:
    i. Average-link similarity for both matrices (GMM-THAC)
    ii. Average-link similarity for the spatial distances and single-link similarity for the temporal distances (GMM-THACmin)

It can be noticed that two variants of the proposed HAC are tested : with and without the temporal distance matrix. As expected, the presence of the temporal distance matrix during the HAC operation improves the result considerably.

The results are shown in the following table:

TABLE I. EXPERIMENT RESULTS

Method	**Score**		
	F-measure	EP_n	GTP_n
EM-GMM	0.7505	0.8539	0.6695
HAC	0.8290	0.7441	0.9357
GMM-HAC	0.8516	0.7528	0.9803
GMM-THAC	**0.8844**	0.8090	0.9753
GMM-THACmin	0.8803	0.8089	0.9655

As can be noticed from the table above, the novel variants of the HAC algorithm significantly improve the efficiency of clustering of the original HAC.

ACKNOWLEDGMENT

A special thanks to my friend Jonathan Strahl for his late night proofreading and careful questioning. It saved a lot of type errors and made me think deeply.

REFERENCES

[1] L. Sirovitch and M. Kirby, “Low-dimensional procedure for the characterization of human faces”, in Journal of Optical Society of America, 4(3):519-524, 1987.

[2] M. Turk and A. Pentland, “Face recognition using eigenfaces”, in Proc. IEEE International Conference on Computer Vision and Pattern Recognition, 586-591, USA, 1991.

[3] R. Gross, J. Shi and J. Cohn, “Quo vadis face recognition?”, in Proc. Third Workshop on Empirical Evaluation Methods in Computer Vision, USA, 2001.

[4] C. Czirjek, N. E. O'Connor, S. Marlow and N. Murphy, "Face detection and clustering for video indexing applications", in Proc. Advanced Concepts for Intelligent Vision Systems, 2-5, Belgium, 2003.

[5] Z. Liu and Y. Wang, "Major cast detection in video using both audio and visual information", in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, 3:1413-1416, USA, 2001.

[6] D. Ramanan, S. Baker and S. Kakade, "Leveraging archival video for building face datasets", in Proc. IEEE International Conference on Computer Vision, 1-8, Brazil, 2007.

[7] T. Hertz, N. Shental, A. Bar-Hillel and D. Weinshall, "Enhancing image and video retrieval: learning via equivalence constraints", in Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2:668-674, USA, 2003.

[8] A. K. Jain and R. C. Dubes, "Algorithms for Clustering Data", Prentice-Hall Inc., USA, 1988.

[9] S. Schwab, T. Chateau, C. Blanc and L. Trassoudaine, “A multi-cue spatio-temporal framework for automatic frontal face clustering in video sequences”, in EURASIP Journal on Image and Video Processing, 2013.

[10] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features”, in Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1:511-518, USA, 2001.

[11] M. Lades et al., "Distortion invariant object recognition in the dynamic link architecture", in IEEE Transactions on Computers, 42(3):300–311, 1993.

[12] J.G. Daugman, "Complete Discrete 2D Gabor Transforms by Neural Networks for Image Analysis and Compression", in IEEE Transactions on Acoustics, Speech and Signal Processing, 36(7):1169-1179, 1988.

[13] C.M. Bishop, “Pattern Recognition and Machine Learning”, Springer-Verlag New York, USA, 2006.

[14] G.E. Schwarz, "Estimating the dimension of a model", in Annals of Statistics, 6(2):461–464, 1978.

[15] S. Kayal, “Face Clustering Experiments on News Video Images”, in Proc. International Conference on Intelligent and Automation Systems, Vietnam, 2013.