Multi-observation Face Recognition in Videos based on Label Propagation

Bogdan Raducanu
Computer Vision Center, Edifici "O" - Campus UAB
01893 Bellaterra (Barcelona), SPAIN
bogdan@cvc.uab.es

Alireza Bosaghzadeh [1], Fadi Dornaika [1,2]
[1] University of the Basque Country UPV/EHU, San Sebastian, Spain
[2] IKERBASQUE, Basque Foundation for Science, Bilbao, SPAIN

Abstract

In order to deal with the huge amount of content generated by social media, especially for indexing and retrieval purposes, the focus shifted from single object recognition to multi-observation object recognition. Of particular interest is the problem of face recognition (used as primary cue for persons' identity assessment), since it is highly required by popular social media search engines like Facebook and Youtube. Recently, several approaches for graph-based label propagation were proposed. However, the associated graphs were constructed in an ad-hoc manner (e.g., using the KNN graph) that cannot cope properly with the rapid and frequent changes in data appearance, a phenomenon intrinsically related with video sequences. In this paper, we propose a novel approach for efficient and adaptive graph construction, based on a two-phase scheme: (i) the first phase is used to adaptively find the neighbors of a sample and also to find the adequate weights for the minimization function of the second phase; (ii) in the second phase, the selected neighbors along with their corresponding weights are used to locally and collaboratively estimate the sparse affinity matrix weights. Experimental results performed on Honda Video Database (HVDB) and a subset of video sequences extracted from the popular TV-series 'Friends' show a distinct advantage of the proposed method over the existing standard graph construction methods.

1. Introduction

Nowadays, the multimedia content that is generated and exchanged through social networks increases at a huge rate. According to [2], 300 hours of video are uploaded to YouTube alone every minute. This huge amount of media requires adequate annotation for efficient indexing and retrieval purposes. It's obviously that this task can't be performed manually. To cope with this constraint, automatic approaches have been proposed which require the weak labeling of the data. According to this concept, only a reduced number of samples is annotated (labeled) and for the remaining ones the annotation is performed automatically, using an algorithm that is able to infer the unknown or missing labels [4]. In terms of pattern recognition, this requires a different matching paradigm in which the focus will shift from single object recognition to multi-observation object recognition. Faces represent a very important and particular class of objects that we primarily use to assess persons' identity. Although state of the art techniques in face recognition can achieve very high accuracy rates under controlled conditions, in unconstrained environments face recognition still remains a challenging problem.

Fortunately, the video data provides a rich and redundant stream of information, which can be exploited to solve the inherent uncertainty of image-based recognition like sensitivity to low resolution, pose variations and occlusion, leading to more accurate and robust recognition. Recently, face recognition have been used for character-based video search in movies [6, 11] or to automatize the process of friends tagging in images shared via social networks [17, 20, 1]. A recent survey on face recognition in videos can be found in [15].

To this end, two categories of approaches were proposed. The first category uses manifold learning paradigms in which the face subspace is constructed using many examples depicting subjects in different poses [19, 23]. The second category generates frontal face from the input image and then apply classic face recognition methods on the reconstructed frontal face image. This category can be split into two main kinds of approaches: i) 3D morphable models [12], and ii) View-based methods [14]. View-based methods train a set of 2D models, each of which is designed to cope with shape or texture variation within a small range of

viewpoints. in which the face subspace is constructed using many examples depicting subjects in different poses.

In [5], the authors propose a Local Linear regression method for pose invariant face recognition. The proposed method can generate the virtual frontal view from a given non-frontal face image. The whole non-frontal face image is partitioned into multiple local patches and then linear regression is applied to each patch for the prediction of its virtual frontal patch. The method requires the pose of the non-frontal pose as input in order to predict the frontal face. Following the approach of Active Appearance Models, [16] develops a face model and a rotation model which can be used to interpret facial features and synthesize realistic frontal face images when given a single novel face image. In [3], the authors address the non-frontal face recognition using morphable models.

In the context of semi-supervised learning, graph-based label propagation can be seen as a powerful tool that solves the multi-observation recognition problem. In [7], the authors proposed a graph-based label propagation method that can infer the labels of unknown observations by optimizing a penalty function based on label consistency. In [9], the authors extended the work of [7] by including the constraint that multiple observations have the same label. However, in both works the graph was constructed in an ad-hoc way, that is, it uses a KNN graph.

In this paper, we propose an extension of our recent work [8] regarding adaptive graph construction. More concrete, our novel approach is based on a two-phase scheme: (i) the first phase is used to adaptively find the neighbors of a sample and also to find the adequated weights for the minimization function of the second phase; (ii) in the second phase, the selected neighbors along with their corresponding weights are used to locally and collaboratively estimate the sparse affinity matrix weights. We use the obtained graph in order to infer the label for multi-observation based face recognition. Experimental results performed on Honda Video Database (HVDB) [10] and a subset of video sequences extracted from the popular TV-series 'Friends' show a distinct advantage of the proposed method over the existing standard graph construction methods.

The paper is structured as follows: Section 2 contains a brief review of existing graph-construction methods. Section 3 presents the proposed algorithm. In Section 4 we report our experimental evaluation. Finally, Section 5 concludes this paper.

2. Related work: graph construction methods

The data graph is a powerful tool that encodes pairwise similarities among data samples. To this end, a weighted graph $G = (V; E; \mathbf{W})$ is constructed, where V denotes the set of N nodes of the graph corresponding to N data samples and $E \subseteq V \times V$ denotes the set of edges between nodes. For undirected graphs, $\mathbf{W}$ is a symmetric non-negative similarity matrix representing the weights of the edges, i.e., node i is connected to node j by an edge whose weight is equal to w_{ij}. An ideal similarity matrix, hence an ideal similarity graph G, is one in which nodes that correspond to points from the same subspace are connected to each other and there are no edges between nodes that correspond to points in different subspaces.

2.1. K Nearest Neighbors graph

The KNN graph is a well known scheme for constructing data graphs. Given N data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^D$, one can build a nearest neighbor graph G to model the local geometrical structure. For each data point $\mathbf{x}_i$, we find its k nearest neighbors and put an edge between $\mathbf{x}_i$ and its neighbors. Let $N(\mathbf{x}_i) = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k\}$ be the set of its k nearest neighbors. Thus, the weight matrix of G can be defined as follows:

$$W_{ij} = \begin{cases} sim(\mathbf{x}_i, \mathbf{x}_j) & \text{if } \mathbf{x}_j \in N(\mathbf{x}_i) \text{ or } \mathbf{x}_i \in N(\mathbf{x}_j) \\ 0, & otherwise \end{cases} \quad (1)$$

where $sim(\mathbf{x}_i, \mathbf{x}_k)$ is a real value that encodes the similarity between $\mathbf{x}_i$ and $\mathbf{x}_k$. Simple choices for this function are the Kernel heat and the cosine.

2.2. LLE graph

Locally Linear Embedding (LLE) [13] focuses on preserving the local structure of data. LLE formulates the manifold learning problem as a neighborhood-preserving embedding, which learns the global structure by exploiting the local linear reconstructions. It estimates the reconstruction coefficients by minimizing the reconstruction error of the set of all local neighborhoods in the dataset. It turned out that linear coding used by LLE can be used for computing the graph weight matrix.

Thus, LLE graph can be obtained by applying two stages: adjacency matrix computation followed by the linear reconstruction of samples from their neighbors. The adjacency matrix can be computed using the KNN or ϵ-Neighborhood method. The non-zero entries of the weight matrix $\mathbf{W}$ are estimated by reconstructing the sample from its neighboring points and minimizing the ℓ_2 reconstruction error defined as

$$\sum_{i=1}^{N} \|\mathbf{x}_i - \sum_j W_{ij}\, \mathbf{x}_j\|^2 \quad s.t. \sum_{j=1}^{N} W_{ij} = 1. \quad (2)$$

where $W_{ij} = 0$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ are not neighbors.

2.3. ℓ_1 graph

Instead of building a graph in two different processes of adjacency construction and weight calculation, the authors

in [22] tried to unify them in one single process. In their proposed method every sample is coded as a sparse linear combination of the rest of the training samples and the contributions of images in representing the sample are considered as weights.

Consider a D dimensional vector $\mathbf{y}$ as an input and a $D \times N$ database matrix $\mathbf{X}$, containing N samples. The goal is to represent input $\mathbf{y}$ as a sparse linear combination of database matrix $\mathbf{X}$. Mathematically, it can be written as

$$\min ||\mathbf{b}||_1 \quad s.t. \quad \mathbf{y} = \mathbf{X}\,\mathbf{b} \tag{3}$$

where vector $\mathbf{b} \in \mathbb{R}^N$ is the coefficient vector. Due to the presence of noise, Eq. (3) will become

$$\min ||\mathbf{b}||_1 \quad s.t. \quad \|\mathbf{y} - \mathbf{X}\,\mathbf{b}\|_2 < \zeta \tag{4}$$

which ζ represents a given tolerance error.

By solving the above minimization problem, the sparse vector $\mathbf{b}$ shows the contribution of each sample in reconstructing the input signal $\mathbf{y}$. As the vector $\mathbf{b}$ is sparse a lot of its elements are zero and few of them have non-zero values. Samples in the database which are far from the input signal will have very small or zero coefficients. The more similar a signal in the database to the sample, the bigger it's coefficient. In this way the neighbors and their weights are calculated simultaneously.

There is also a formulation that can account for sparse outliers in the signals. This is given by:

$$\min ||\mathbf{b}||_1 + ||\mathbf{e}||_1 \quad s.t. \quad \mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{e} \tag{5}$$

The above problem can be casted into the form given in (3) by solving for the augmented vector $\mathbf{b}' = (\mathbf{b}^T, \mathbf{e}^T)^T$:

$$\min \|\mathbf{b}'\|_1 \quad s.t. \quad [\mathbf{X}\ \mathbf{I}]\,\mathbf{b}' = \mathbf{y} \tag{6}$$

Therefore, by using the above coding for every training sample $\mathbf{x}_i$ and calculating sparse vector $\mathbf{b}_i$, one can construct the $\mathbf{W}$ matrix using the set of computed vectors $(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_N)$. A directed graph with an asymmetric weight matrix $\mathbf{W}$ can be constructed using two following formula:

$$W_{ij} = |b_i(j)| \tag{7}$$

3. Proposed graph construction

As explained in the previous section there are many different methods to build a graph. Our objective is to provide an efficient tool for graph construction that has the same advantages of ℓ_1 graphs. In our proposed method, we construct the graph of a database by directly using the coding of any training image with respect to the rest of the set. We were inspired by recent advances in collaborative coding, namely the Weighted Regularized Least Square minimization method (WRLS) proposed in [21]. In this work, the authors proposed a linear coding scheme in order to classify samples according to the collaborative reconstruction error. Their proposed criterion is based on the sum of three parts: (i) L_2 norm of the reconstruction error, (ii) a regularization term set to the L_2 norm of the coefficients vector, (iii) a weighted sum of the squared coefficients. Since the weights are set to the distances between the test sample and the training samples, a kind of sparsity is included in the global criterion.

3.1. Weighted Regularized Least Square minimization

Assume we have a datum $\mathbf{y}$ and want to represent it by a linear combination of samples (or subset) of the dataset $\mathbf{X}$ as $\mathbf{y} = \mathbf{X}\,\mathbf{b}$, where $\mathbf{b}$ is a N dimensional vector containing the weights of all samples in the database in representing $\mathbf{y}$ and $\mathbf{X} = [\mathbf{x}_1\mathbf{x}_2...\mathbf{x}_N]$ ($D \times N$ matrix) is the data matrix. We assume that each sample $\mathbf{x}_i$ is normalized using its ℓ_2 norm. The general formula to solve the above equation will be:

$$\min ||\mathbf{b}||_r \quad s.t. \quad \|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2 < \epsilon \tag{8}$$

which tries to represent the datum $\mathbf{y}$ by the smallest $||\mathbf{b}||_r$ which represents the ℓ_r-norm of $\mathbf{b}$ (i.e., $||\mathbf{b}||_r = (\sum_j |b(j)|^r)^{1/r}$.

In our work, we use the ℓ_2 norm of residual error with sparsification. Our sparsification consists in two main modules. The first one uses a weighted regularization in which each unknown coefficient has an independent weight derived from the similarity between the test sample and the corresponding sample in the dataset. The second module uses a two phase WRLS where the second stage only uses samples having large coefficients and adaptively chosen without any predefined parameter. Unlike the ℓ_1 minimization, with the ℓ_2 norm, Eq. (8) will have a closed form solution and it can be calculated in a more efficient way.

The unknown vector $\mathbf{b}$ can be calculated by minimizing the following criterion (WRLS):

$$\mathbf{b} = \arg\min_{\mathbf{b}} \frac{1}{2} \left(\|\mathbf{y} - \mathbf{X}\mathbf{b}\|_2^2 + \sigma \sum_{j=1}^{N} p_j^2\, b_j^2 \right) \tag{9}$$

where p_j is a positive weight associated with example $\mathbf{x}_j$ (or equivalently b_j). The solution to Eq. (9) will reconstruct the input signal $\mathbf{y}$ by a combination of the space spanned by $\mathbf{X}$. In Eq. (9), the criterion has two terms: the reconstruction error and the weighted regularized term. Thus, σ is a small positive scalar that balances the two terms effect.

By having different values for p_j one can control the coefficients getting bigger or smaller. The bigger the p_j is the

smaller the b_j would be and vice versa. By using simple linear algebra calculations, the solution to Eq. (9) has a closed form solution that is given by:

$$\mathbf{b} = \left(\mathbf{X}^T\mathbf{X} + \sigma\,\mathbf{P} \right)^{-1} \mathbf{X}^T\,\mathbf{y} \tag{10}$$

$\mathbf{P}$ is a diagonal matrix with elements $\mathbf{P}_{jj} = p_j$. In our work, we use the following formula:

$$p_j = 1 - \exp(-\|\mathbf{y} - \mathbf{x}_j\|^2) \tag{11}$$

3.2. Two Phase WRLS (TPWRLS)

We have seen above that the weighted regularization term imposes a kind of sparsification on the coefficients b_j due to the use of the weights. In this section, we will propose an additional sparsification module that acts by invoking two passes of the WRLS minimization process. The first pass considers the whole data as a dictionary for WRLS minimization. Once the vector of coefficients is estimated, a subset of examples will be selected from the original dictionary and used as the new dictionary for a second pass of WRLS. The selection will rely on the magnitude of the coefficients.

Once the first pass is achieved, we have an additional information provided by the estimated coefficients b_j. Indeed, $|b_j|$ provides a reliable similarity measurement between the sample $\mathbf{y}$ and the example $\mathbf{x}_j$. Thus, our intuition is to keep the most similar examples and remove the remaining ones by exploiting the calculated vector $\mathbf{b}$. To do so, we compute a threshold given by the mean of similarities: $TH(\mathbf{y}) = \frac{1}{N}\sum_{j=1}^{N} |b_j|$.

Let $\mathbf{X}_s$ be the data matrix formed by the selected examples (the ones whose $|b_j|$ is greater than the mean similarity). Then, the vector $\mathbf{b}'$ associated with the selected examples will be solved using a formula similar to Eq. (10):

$$\mathbf{b}' = \left(\mathbf{X}_s^T\mathbf{X}_s + \sigma\,\mathbf{P}' \right)^{-1} \mathbf{X}_s^T\,\mathbf{y} \tag{12}$$

where the diagonal weights p'_j are given by:

$$p'_j = \frac{1}{|b_j|} \tag{13}$$

In order to avoid a very high disparity among the obtained p'_j weights, we normalized them using a unit variance normalization scheme. The above weights can reinforce the sparsification of the new estimated coefficients. Furthermore, the size of the data matrix $\mathbf{X}_s$ in the second phase is smaller than that of the whole data matrix $\mathbf{X}$ and, hence, the coefficients obtained in the second phase are sparser in comparison to those obtained in the first phase with the whole database. In the TPWRLS, the original N-vector $\mathbf{b}$ is set as follows. A non-selected example $\mathbf{x}_j$ will have $b_j = 0$ and a selected one will have the corresponding coefficient in the vector $\mathbf{b}'$ estimated by Eq. (12).

The detailed procedure for the TPWRLS graph construction is listed in Algorithm 1. This algorithm estimates the i^{th} row of the affinity matrix by coding the sample $\mathbf{x}_i$ w.r.t. to the set $\mathbf{X} - \{\mathbf{x}_i\}$. Note that the constructed graph is a directed graph, i.e., the weight matrix $\mathbf{W}$ is asymmetric.

Regarding the sample-based threshold used by TPWRLS, in theory any value that belongs to $[0, |b_{max}|]$ can be used. However, we have empirically found that the mean and the median provide good performances (as it will be shown next). However, we found that the mean has provided slightly better results than the median. The mean has not suffered from large variations on the coefficients since the data samples (the atoms of dictionary) were normalized to unit vectors. The use of median has provided lower thresholds and thus allowing larger dictionary in the second stage.

3.3. Difference between TPWRLS graph and existing graphs

The main differences between the proposed TPWRLS graph and existing graphs are as follows:

- The proposed TPWRLS does not need any predefined neighborhood size since it is based on self-representativeness of data. Therefore, it adapts to sample distributions.
- The closeness among data samples is implicitly incorporated by adopting weighting schemes as well as a second stage that produces a small subset of samples having the highest coding coefficients.
- The weights in the second stage are provided by a coding scheme and not set to similarities in the original space of data such as the Euclidean distance, cosine score, and Gaussian kernel.
- Compared to ℓ_1 graph, the proposed TPWRLS provides an edge weights coding scheme on a cleverly selected set of samples.

3.4. Multi-observation Recognition Based on Label Propagation

We assume that we have N labeled samples belonging to C classes. We also assume that we have r unlabel samples having the same unknown label. The objective is to infer the unknown label via label propagation over the graph constructed using both labeled and unlabeled samples. This is schematically illustrated in Figure 1.

Let $\mathbf{y}_i$ be the posterior probability of samples belonging to C different classes, namely, $y_i(c) = p(c|\mathbf{x}_i); c = 1, 2, \ldots, C$. Let $\mathbf{W}$ be the similarity matrix associated with the graph associated with the $N + r$ samples. For a labeled sample $\mathbf{x}_i$, $y_i(c) = 1$ if $\mathbf{x}_i$ belongs to the c^{th} class; $y_i(c) =$

Algorithm 1 TPWRLS graph construction

Data: A dataset $\mathbf{X}$
Result: A weight matrix $\mathbf{W}$ of its graph
for $i = 1$ **to** N **do**
  * Pick the sample $\mathbf{x}_i$ and set $\mathbf{X}'$ = $\mathbf{X}$ - $\{\mathbf{x}_i\}$
  * Compute the $(N-1) \times (N-1)$ diagonal matrix $\mathbf{P}$ using Eq. (11)
  * Calculate the $(N-1) \times 1$ vector $\mathbf{b}$ as $\mathbf{b} = (\mathbf{X}'^T \mathbf{X}' + \sigma \mathbf{P})^{-1} \mathbf{X}'^T \mathbf{x}_i$
  * Compute a threshold for $\mathbf{x}_i$ as $TH(\mathbf{x}_i) = \frac{1}{N-1} \sum_{j=1}^{N-1} |b_j|$
  * Set the selected samples $\mathbf{X}_s$ (whose $|b_j|$ are above that threshold)
  * Form the new diagonal weight matrix $\mathbf{P}'$ using Eq. (13)
  * Calculate the vector $\mathbf{b}'$ as $\mathbf{b}' = \left(\mathbf{X}_s^T \mathbf{X}_s + \sigma \mathbf{P}_s \right)^{-1} \mathbf{X}_s^T \mathbf{x}_i$
  * Set the sparse vector $\mathbf{b}$ from $\mathbf{b}'$
  for $j = 1$ **to** N **do**
    $W_{ij} = |b_j|$
  end for
end for

0, otherwise. Consider the data matrix $\mathbf{X} = (\mathbf{X}_l, \mathbf{X}_u)$ (a $D \times (N+r)$ matrix) containing labeled and unlabeled data, and its associated label matrix $\mathbf{Y} = (\mathbf{Y}_l, \mathbf{Y}_u)$ (a $C \times (N+r)$ matrix) where N denotes the number of labeled samples and r the number of unlabeled samples.

The problem of label propagation [24] is to infer the label matrix $\mathbf{Y}_u$ given the whole data $\mathbf{X} = (\mathbf{X}_l, \mathbf{X}_u)$ and the known label matrix $\mathbf{Y}_l$. This can be achieved by minimizing the following criterion:

$$\min E(\mathbf{Y}) = \sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij} \tag{14}$$

An explanation of this objective is as follows. When the samples $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar, namely, the graph weight W_{ij} is large, the distance between $\mathbf{y}_i$ and $\mathbf{y}_j$ should be small in order to minimize the objective, namely the class information of the sample $\mathbf{x}_i$ and $\mathbf{x}_j$ should be similar.

The objective can be further rewritten as

$$\begin{aligned} E(\mathbf{Y}) &= \sum_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|^2 W_{ij} \\ &= trace(\mathbf{Y}\,\mathbf{D}_{row}\,\mathbf{Y}^T + \\ &\quad \mathbf{Y}\,\mathbf{D}_{col}\,\mathbf{Y}^T - \mathbf{Y}\,\mathbf{W}\,\mathbf{Y}^T - \mathbf{Y}\,\mathbf{W}\,\mathbf{Y}^T) \\ &= trace(\mathbf{Y}\,\mathbf{L}\,\mathbf{Y}^T) \end{aligned} \tag{15}$$

where $\mathbf{L}$ is given by $\mathbf{L} = \mathbf{D}_{row} + \mathbf{D}_{col} - (\mathbf{W} + \mathbf{W}^T)$. It is obvious that the matrix $\mathbf{L}$ is symmetric.

$\mathbf{D}_{row}$ is a diagonal matrix whose diagonal elements are the row sums of the corresponding rows of $\mathbf{W}$, and $\mathbf{D}_{col}$ is a diagonal matrix whose diagonal elements are the column sums of the corresponding columns of $\mathbf{W}$. $\mathbf{D}_{row} - \mathbf{W}$ and $\mathbf{D}_{col} - \mathbf{W}^T$ are the row and column Graph Laplacian matrices respectively.

Since the r samples have the same unknown label, the unknown label matrix $\mathbf{Y}_u$ will have C configurations $(\mathbf{Y}_u(1), \ldots, \mathbf{Y}_u(C)$ where $\mathbf{Y}_u(c)$ has only the c^{th} row equal to ones and the the rest of the rows are zeros. Therefore, the whole label matrix $\mathbf{Y} = (\mathbf{Y}_l, \mathbf{Y}_u)$ can be written as $\mathbf{Y} = (\mathbf{Y}_l, \mathbf{Y}_u(c))$ where $\mathbf{Y}_l$ is constant. To infer the label of the unknown observations $\mathbf{X}_u$, the following formula can be used:

$$c^\star = \arg\min E(\mathbf{Y}_c) \tag{16}$$

where $\mathbf{Y}(c) = (\mathbf{Y}_l, \mathbf{Y}_u(c))$. Thus, the optimal label is inferred using C evaluations of the term $E(\mathbf{Y}_c)$.

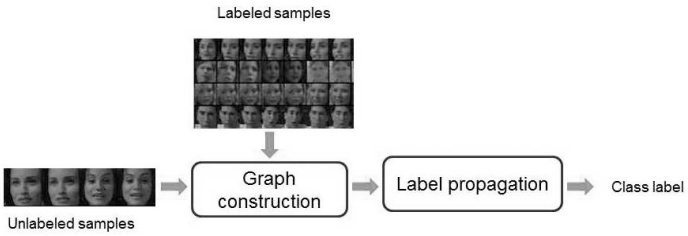

Figure 1. Label propagation process

4. Experimental Results

In order to test our proposed approach for graph construction, we applied it to the problem of multi-observation based face recognition in videos. Starting with a small training dataset, the goal was to automatically assign class labels to a sequence of frames (of varying sizes) in order to simulate the automatic annotation of video streams. For this purpose, we used two video datasets. The first one, is the public Honda Video Database (HVDB), while the second one is a custom dataset, created from the popular 'Friends' TV-series. In the first case, faces has been manually extracted, while for the second one faces were automatically extracted using the Viola-Jones face detector [18]. We aim to test our algorithm on two situations: (i) one, in which we

have a relative high number of classes, but a relatively small number of samples per class (HVDB) and (ii) another one, in which we have a relative small number of classes but a relative high number of samples per class ('Friends' dataeset). Another difference resides in the following fact: in HVDB, people are showing exaggerated head movements, under unconstrained illumination conditions, since its main purpose was to offer a benchmark for face tracking algorithms; in 'Friends', people show a normal behavior, since they are acting in a more realistic scenario.

The evaluation methodology used was the same for both datasets. Our proposed approach, TPWRLS, was compared with some of the competing approaches for graph construction, namely: K-Nearest Neighbors, LLE, ℓ_1 graph and WRLS.

4.1. Honda Video Database

This database consists of 22 persons with approximately 100 images per person. Some instances of this dataset are depicted in figure 2. The experimental results are depicted in table 1. We divided the data in 10 random splits for training/testing. We have considered three modes: $10\% - 90\%$, $20\% - 80\%$ and $30\% - 70\%$. Regarding the test data, it has been divided in monotonic chunks (all samples in a class belong to the same person) with lengths of 3, 4 and 5. The 'r' parameter corresponds to chunk size.

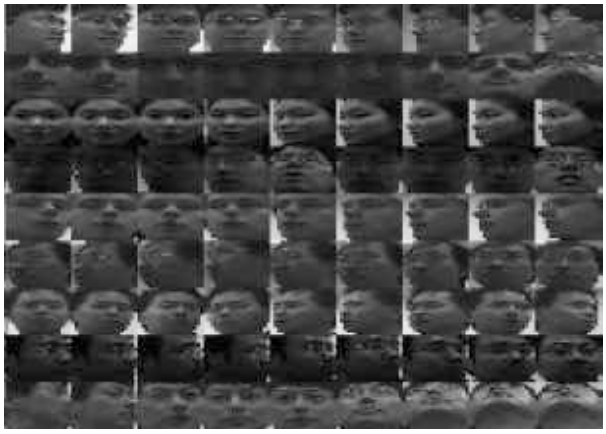

Figure 2. Some samples from the HVDB database

As it can be appreciated, the proposed TPWRLS graph construction method outperforms the competing ones.

4.2. Friends dataset

This dataset consists of 6 persons with approximately 1000 images per person. Some instances of this dataset are depicted in figure 3. The experimental results are depicted in table 2. We followed a similar protocol as in the HVDB case: we divided the data in 10 random splits for training/test. But, the difference here is that we have considered only one mode: $5\% - 95\%$, due to the fact that we had significant number of samples per class. Regarding the test data, it was divided in monotonic chunks (all samples in a class belong to the same person) with lengths of 5, 10, 15 and 20. The 'r' parameter corresponds to chunk size.

Figure 3. Some samples from the Friends dataset

As in the case of HVDB, with Friends dataset TPWRLS outperforms the existing graph construction methods. In figure 4 we depict a graphical representation of the results summarized in table 2.

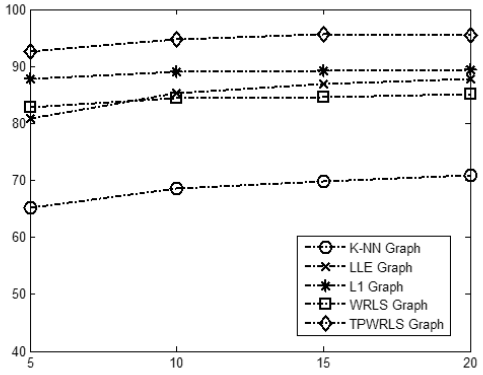

Figure 4. Recognition rate based on all five graph construction methods ('Friends' dataset)

As expected, for both databases, the performance improves when either the chunk length increases or the training set size increases. For the second dataset, we can observe that a larger number of observations has not led to significant improvements.

Algorithms' complexity. We have measured the computing time of the proposed graph construction method. To this end, we used the training percentage of 10% together with $r = 3$ images. In other words, the graph size is 234 images. We used a non-optimized MATLAB code running on a PC equipped with an Intel XEON E5 CPU at 3.5 Ghz. Table 3 summarizes the performance of all the five methods for graph construction. The first column depicts the CPU

time associated with the graph construction stage, the second column depicts the CPU time associated with the recognition step (label propagation), and the third column depicts the recognition rate. We can appreciate that even TPWRLS is not the fastest graph construction method, it is definitely the best one in terms of recognition accuracy.

5. Conclusions and Future Work

In this paper we have presented a novel approach for graph construction. The proposed approach have been used to perform label propagation for multi-observation based face recognition. The experimental results performed on two datasets demonstrated the superiority of our approach with respect to other graph-construction methods. This overall application presented, label propagation for multi-observation based recognition, could be very useful for annotating the huge amount of video data generated by the social media. In the future, we are planing to extend the current work, by performing multi-observation based manifold incremental learning. In this case, due to the large number of quasi-similar views contained in the multi-observation chunk, another problem would be to identify the most relevant samples that could be used subsequently for manifold updating.

Acknowledgments B. Raducanu is supported by the projects TIN2013-41751 and TIN2013-49982-EXP, MINECO, Spain.

References

[1] The face detection algorithm set to revolutionize image search. http://www.technologyreview.com/view/535201/the-face-detection-algorithm-set-to-revolutionize-image-search/. 1

[2] Youtube statistics. http://www.youtube.com/yt/press/statistics.html. 1

[3] V. Blanz, P. Grother, P. J. Phillips, and T. Vetter. Face recognition based on frontal views generated from non-frontal images. In *Proc. of IEEE CVPR 2005*, pages 454 – 461, San Diego (CA), USA, 2005. 2

[4] P. Bojanowski, R. Lajugie, F. Bach, I. Laptev, J. Ponce, C. Schmid, and J. Sivic. Weakly supervised action labeling in videos under ordering constraints. In D. F. et al., editor, *Computer Vision – ECCV 2014*, volume 8693 of *LNCS*, pages 628–643, 2014. 1

[5] X. Chai, S. Shan, X. Chen, and W. Gao. Locally linear regression for pose-invariant face recognition. *IEEE Trans. on Image Processing*, 16(7):1716–1725, 2007. 2

[6] R. G. Cinbis, J. Verbeek, and C. Schmid. Unsupervised metric learning for face identification in tv video. In *Proc. of IEEE ICCV 2011*, pages 1559 – 1566, Barcelona, Spain, 2011. 1

[7] T. N. L. J. W. D. Zhou, O. Bousquet and B. Scholkopf. Adv. neural inf. process. syst. In *Learning with local and global consistency*, 2004. 2

[8] F. Dornaika, A. Bosaghzadeh, and B. Raducanu. In A. A. S. et al., editor, *Human Behavior understanding*, volume 8212 of *LNCS*, pages 124–135, 2013. 2

[9] E. Kokiopoulou and P. Frossard. Graph-based classification of multiple observation sets. *Pattern Recognition*, 43(12):3988–3997, 2010. 2

[10] K.-C. Lee, J. Ho, M.-H. Yang, and D. Kriegman. Visual tracking and recognition using probabilistic appearance manifolds. *Computer Vision and Image Understanding*, 99:303–331, 2005. 2

[11] E. G. Ortiz, A. Wright, and M. Shah. Face recognition in movie trailers via mean sequence sparse representation-based classification. In *Proc. of IEEE CVPR 2013*, pages 3531 – 3538, Portland (OR), USA, 2013. 1

[12] A. Patel and W. Smith. 3d morphable face models revisited. In *Proc. of IEEE CVPR 2009*, pages 1327–1334, Miami (FL), USA. 1

[13] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. 2

[14] P. Sauer, T. Cootes, and C. Taylor. Accurate regression procedures for active appearance models. In *Proc. of BMVC 2011*, pages 1–8, Dundee, UK, 2011. 1

[15] C. Shan. Face recognition and retrieval in video. In D. S. et al., editor, *Video Search and Mining*, volume 287 of *SCI*, pages 235–260, 2010. 1

[16] T. Shan, B. C. Lovell, and S. Chen. Face recognition robust to head pose from one sample image. In *Proc. of IEEE ICPR 2006*, pages 515–518, Hong Kong, P.R. China, 2006. 2

[17] Z. Stone, T. Zickler, and T. Darrell. Autotagging facebook: Social network context improves photo annotation. In *Proc. of IEEE CVPR Workshops*, pages 1–8, Anchorage (AK), USA, 2008. 1

[18] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004. 5

[19] R. Wang, S. Shan, X. Chen, Q. Dai, and W. Gao. Manifold-manifold distance and its application to face recognition with image sets. *IEEE Trans. on Image Processing*, 21(10):4466–4479, 2012. 1

[20] X. Wang, C. Zhang, and Z. Zhang. Boosted multi-task learning for face verification with applications to web image and video search. In *Proc. of IEEE CVPR 2009*, pages 142–149, Miami (FL), USA, 2009. 1

[21] J. Waqas, Z. Yi, and L. Zhang. Collaborative neighbor representation based classification using l_2-minimization approach. *Pattern Recognition Letters*, 34(2):201–208, 2013. 3

[22] S. Yan and H. Wang. Semi-supervised learning by sparse representation. In *SIAM International Conference on Data Mining*, pages 792–801, 2009. 3

[23] H. Zhang, Y. Zhang, and T. S. Huang. Pose-robust face recognition via sparserepresentation. *Pattern Recognition*, 46:1511–1521, 2013. 1

[24] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proc. of IEEE ICML 2003*, pages 1–8, Washington DC, USA, 2003. 5

Split \ Method	K-NN graph	LLE graph	ℓ_1 graph	WRLS graph	TPWRLS graph
10%-90%					
r=3	71.30	80.89	82.42	72.37	**86.07**
r=4	75.33	83.40	84.75	79.52	**88.72**
r=5	78.46	87.38	87.27	81.56	**91.38**
20%-80%					
r=3	85.22	89.55	91.55	90.02	**95.62**
r=4	88.79	91.33	93.13	91.66	**97.77**
r=5	90.36	92.54	93.88	91.59	**98.10**
30%-70%					
r=3	88.61	92.04	92.25	91.16	**98.17**
r=4	91.24	93.36	94.74	92.74	**98.98**
r=5	92.68	94.23	95.75	93.04	**98.93**

Table 1. Multi-observation recognition rate on Honda Video Database

Split \ Method	K-NN graph	LLE graph	ℓ_1 graph	WRLS graph	TPWRLS graph
5%-95%					
r=5	65.15	80.78	87.85	82.85	**92.66**
r=10	68.54	85.29	89.10	84.38	**94.73**
r=15	69.80	87.00	89.17	84.65	**95.58**
r=20	70.82	87.81	89.42	85.21	**95.52**

Table 2. Multi-observation recognition rate on Friends Dataset

	Graph construction (CPU time)	Recognition (CPU time)	Recognition rate
KNN graph	0.8 (seconds)	0.004 (seconds)	71.30
LLE graph	0.36 (seconds)	0.004 (seconds)	80.89
ℓ_1 graph	2.28 (seconds)	0.004 (seconds)	82.42
WRLS graph	0.56 (seconds)	0.004 (seconds)	72.37
TPWRLS graph	0.7 (seconds)	0.004 (seconds)	86.07

Table 3. CPU time associated with all five graph construction methods for 234 images with a chunk size is r=3.