Neurocomputing 152 (2015) 316–321

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Automatic face annotation in TV series by video/script alignment

Yifan Zhang [a], Zhiqiang Tang [a], Chunjie Zhang [b], Jing Liu [a,*], Hanqing Lu [a]

[a] National Lab of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[b] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100049, China

ARTICLE INFO

Article history:
Received 4 September 2014
Received in revised form
10 October 2014
Accepted 23 October 2014
Communicated by X. Li
Available online 13 November 2014

Keywords:
Face annotation
Face naming
Video analysis
Sequence alignment

ABSTRACT

This paper describes a method for automatically tagging the names to the faces which are collected from uncontrolled TV series videos. The detected faces are firstly partitioned into several clusters. Then we construct a face sequence based on their occurrence order in the video and denote them by cluster labels. It can be assumed that the temporal distribution of the faces in the video roughly follows the temporal distribution of the names in the script. Hence, we propose to annotate the faces by video/script alignment. A global sequence alignment algorithm is employed to find the most probable faces in the face sequence matching to the names in the name sequence. The novelty lies in that we consider the temporal order relationship of the faces and names over the whole video and directly align two heterogeneous sequences. Experiments on real-world videos have demonstrated the effectiveness and efficiency of the proposed method.

1. Introduction

This paper investigates the problem of automatically annotating the faces with the character names in TV series videos. The objective is to find all the faces of certain person and to attach the correct name to them. Based on our work, one can easily use the character name as a query to select the characters of interest and view the related video clips. This character-centric browsing is able to provide a new way for video summarization and digestion, thus bringing us a new viewing experience.

The face annotation task is challenging due to high variability of the faces in pose, scale, facial expression, as well as illumination, occlusion and camera movement. In addition, we also have to face the weakness and ambiguity of the available textual information. Since the text is not well temporally aligned with the video, the crux is how to exploit the relationship between the video and the associated text to build the links between the faces and the names.

The previous efforts on video face annotation can be roughly divided into two categories: classification based and clustering based. Most of the previous works focus on person classification. Everingham et al. [1] combine facial and clothing features to train a Bayesian classifier. Sivic et al. [2] use the facial feature to train a multiple kernel SVM classifier. Tapaswi et al. [3] model each TV series episode as a Markov Random Field, integrating face, clothing and speaking features together to increase the coverage of the labeled persons. Bäuml et al. [4] employ multinomial logistic regression classifiers for multi-class classification which incorporates labeled and unlabeled data. Albeit the classification methods can achieve good performance, they typically rely on the quality of the annotated training exemplars for each person. The training data is obtained either by manual labeling or by textual alignment between subtitles and scripts [1]. Everingham et al. [1] align the subtitles with the scripts by dynamic time warping to get the time tags of the names, and then link them to the faces in the videos to get labeled face exemplars. However, the name cues from the textual alignment are weak and ambiguous since faces of speakers may not be visible, or there may be more than one face visible at the same time. In addition, its application scope is limited by the availability of the subtitle, which usually does not exist in many non-European language TV shows [5].

To extend the application scope to the scenarios when subtitles are not available, some researchers investigate the problem on how to build the linking between the faces and the names without using time tags. These works are mainly based on face clustering. Some sophisticated face clustering methods are proposed. Wu et al. [6] use a Markov random field to model the relationships between faces and incorporate pairwise constraints. Lu and Ip [7] propose a constrained spectral clustering. Both of the methods can propagate the constraints to neighboring data based on smooth assumption. Cinbis et al. [8] propose an unsupervised metric learning algorithm for face identification in TV video. All these methods focus on face clustering, they cannot automatically link the faces to their real names. To this end, in [9], we propose a global face–name matching framework, in which the weighted face graph from the video is matched with the weighted name

* Corresponding author.
E-mail addresses: yfzhang@nlpr.ia.ac.cn (Y. Zhang), zqtang2013@gmail.com (Z. Tang), cjzhang@jdl.ac.cn (C. Zhang), jliu@nlpr.ia.ac.cn (J. Liu), luhq@nlpr.ia.ac.cn (H. Lu).

http://dx.doi.org/10.1016/j.neucom.2014.10.062
0925-2312/

graph from the script. The weighted face graph is constructed by face clustering. Sang and Xu [10] extend the work of [9] by using an ordinal graph and employing a new graph matching algorithm called Error Correcting Graph Matching. Liang et al. [11] propose a generative model in which character histogram is used to depict the correspondence between the video and the script. The face–name association matrix is automatically learned as the parameters of the model. Generally, most of these methods match the faces and names based on local face–name relationship within each scene of a TV series video or a movie. The global temporal order relationship over the whole video is not used. The method in [11] uses forward traversing and backward tracing algorithms to conduct the matching over the whole video. However, it employs the character histogram to compare the faces and names in shots and scenes. Consequently, most of their performance relies on the scene segmentation results.

In this paper, we present a novel framework (see Fig. 1) for face annotation based on face clustering and global video/script alignment. In the framework, global temporal order relationship over the whole video is fully exploited, without relying on scene segmentation. We firstly build two sequences from the video and the script respectively: a face sequence and a name sequence. In the face sequence, faces are denoted by their cluster labels which are obtained by a clustering algorithm. It can be assumed that the temporal distribution of faces of one character is approximately similar to the temporal distribution of their names along the time line. Hence, the face naming problem can be transformed to a sequence alignment problem. In our previous work [12], we use the Levenshtein distance [13] measurement to find the optimal alignment between the face track sequence and the name sequence. Since the Levenshtein distance is the minimum editing distance between two homogeneous sequences, to extend it to two heterogeneous sequences, we have to enumerate all possible matching results between the face clusters and the names. It quickly becomes computationally intractable when the number of the different elements in the two sequences increases. In this paper, we propose a novel method to directly align the two heterogeneous sequences. We use the symmetric K–L divergence to measure the similarity of the elements in the two heterogeneous sequences. Based on the similarity matrix, the global sequence alignment is accomplished by a dynamic programming method called the Needleman–Wunsch algorithm [14], by which the computational complexity is dramatically reduced.

2. Face detection and track linking

The faces in the videos are detected by a cascade object detector in the Computer Vision System Toolbox of MATLAB. We divide the video into shots based on the differences of HSV features between neighboring frames. The faces which are continuous in position and scale within a shot are connected as a face track. The tracks whose average sizes of the bounding boxes are less than 55×55 pixels are regarded as false detection and deleted. Since faces in a track in neighboring frames are usually similar, we uniformly sample one face in every five consecutive frames in a track to reduce the data volume.

A facial landmark detector [15] is employed to detect five facial landmarks on each face. Based on the landmark points, we rectify the original detected faces to a canonical pose with a normalized distance of 30 pixels between two eyes. The facial landmarks and face rectification are illustrated in Fig. 2. On the rectified and cropped face image, we extract the gray level feature from a pixels patch centered by each landmark point (i.e. 40×30, 30×40,

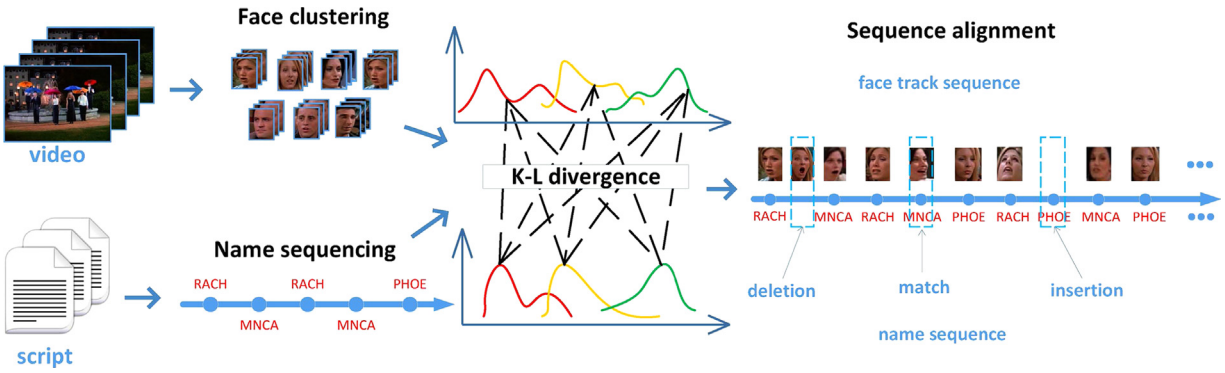


Fig. 1. Framework of face annotation based on face clustering and video/script alignment.

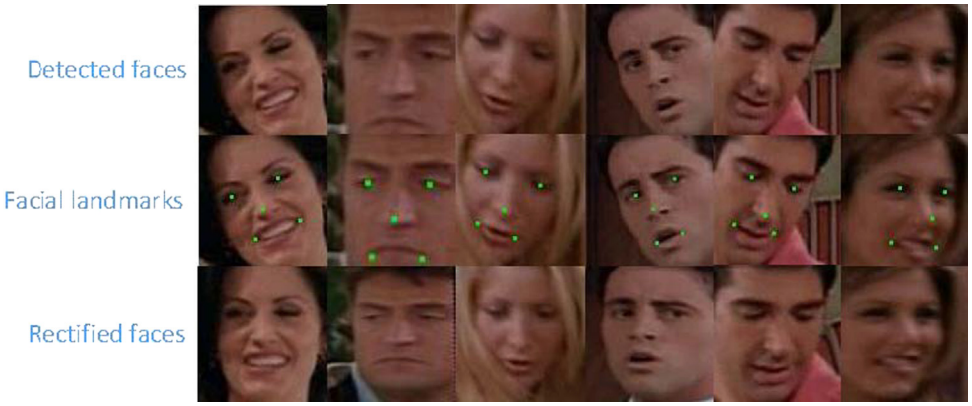


Fig. 2. Face rectification by the facial landmarks. The three rows from top to bottom are the original detected faces, the detected faces with landmarks and the rectified faces.

30×30 pixels for eye, nose and mouth corner landmark points, respectively). Then the five gray level feature vectors are connected together to represent the face.

3. Face track clustering

Supposing that we have obtained the face tracks $X = \{x_1, x_1, \ldots, x_n\}$ from K characters in one video, we want to partition them into L clusters, denoted by $\{1, 2, \ldots, L\}$, where $L \geq K$. It is worth noting that the number of the clusters is not restricted to be the same with the number of the characters. We can increase the number of the clusters to improve the purity for each cluster. For face track clustering, a constrained spectral clustering algorithm [16] is employed. Firstly, we build a similarity graph based on the adjacent distance matrix. The distance between two face tracks is measured by the minimum distance of any pair of two face images in the two tracks. As we know, the two temporally overlapping face tracks in the video must belong to two different persons. The spectral clustering algorithm can incorporate such pairwise constraints during clustering by adjusting the graph similarity matrix. Then the adjusted graph similarity matrix is used in a traditional spectral clustering algorithm. By computing the graph Laplacian matrix and first-k eigenvectors, it can transform the original feature space into a new space. Finally, k-means is used to cluster the data in the new feature space.

4. Global sequence alignment

We extract the face tracks in the video and the names in the script and build two sequences. The two heterogeneous sequences are directly aligned to achieve face–name association. The faces can be labeled with the names aligned to them.

We have gotten a face track sequence $X = x_1 x_2 \cdots x_n$ with each $x_i \in A = \{1, 2, \ldots, L\}$ where A is the face cluster label set, and a name sequence $Y = y_1 y_2 \cdots y_m$ with each $y_j \in B = \{1, 2, \ldots, K\}$, where B is the name set. It is worth noting that L is usually larger than K. To find the optimal alignment between X and Y, we employ a global sequence alignment method described as follows:

We firstly specify a similarity matrix $S \in \mathcal{R}^{L \times K}$ between the face clusters and the names, where L is the number of the face clusters and K is the number of the distinct names in the script. The entry of S, denoted as $S(i,j)$, shows how likely the ith face track cluster should be assigned to the jth name. We use the symmetric Kullback–Leibler divergence to measure the similarity:

$$S(i,j) = \exp\left(-\tfrac{1}{2}\left(D_{KL}(P_i||Q_j) + D_{KL}(Q_j||P_i)\right)\right) \tag{1}$$

where

$$D_{KL}(P_i||Q_j) = \sum_k \ln\left(\frac{P_i(k)}{Q_j(k)}\right) P_i(k) \tag{2}$$

$D_{KL}(\cdot||\cdot)$ is the Kullback–Leibler divergence, P_i and Q_j are the temporal distribution of the ith face track cluster and the jth name along the time line respectively. The temporal distribution is approximated as follows: we uniformly divide the face track sequence and the name sequence into the same number of bins. Each bin records the accumulated number of the faces or the names falling into it. Thus we can get a histogram for any face cluster or name. The temporal distribution is represented by the normalized histogram. After the similarity matrix S is calculated, we normalized it by row to make the maximum value in each row to be 1.

The sequence alignment is accomplished by the Needleman–Wunsch algorithm [14] which is previously used in bioinformatics to align protein or nucleotide sequences. Since the lengths of the two sequences are not the same, the "delete" and "insert" actions will be taken to certain elements in the sequence. Accordingly, a gap penalty d will be given to these two types of actions. Empirically, we set d to be half of the median of the entries less than 1 in the similarity matrix S. To find the alignment with the highest score, a score matrix $G \in \mathcal{R}^{(n+1)\times(m+1)}$ is allocated, where n and m are the lengths of sequence X and Y, respectively. The entry $G(i,j)$ is set to be the maximum score for the optimal alignment till the ith face track in X and the jth name in Y. The calculation of $G(i,j)$ is based on the principle as follows:

$$G(i,j) = \max(G(i-1,j-1) + S(X_i, Y_j), G(i,j-1) + d, G(i-1,j) + d) \tag{3}$$

where the term $G(i-1,j-1) + S(X_i, Y_j)$ is the score of the "match" action. "Match" means the face X_i and name Y_j are linked. The score is calculated by the last step score $G(i-1,j-1)$ plus the similarity between X_i and Y_j. The terms $G(i,j-1)+d$ and $G(i-1,j)+d$) are the scores of the "insert" and "delete" actions, respectively. The two actions mean that one element aligns to a gap in the other sequence. Hence the gap penalty d is added.

The initialization of the process is

$$G(0,j) = d \times j, \quad G(i,0) = d \times i \tag{4}$$

The process of computing the score matrix G is summarized in Algorithm 1.

Algorithm 1. Algorithm for computing the alignment score matrix G.

Input: Face track sequence X, name sequence Y, similarity matrix S
Output: alignment score matrix G

for do $i = 0$ to length(X)
  $G(i,0) \leftarrow d \times i$
end for
for do $j = 0$ to length(Y)
  $G(0,j) \leftarrow d \times j$
end for
for do $i = 1$ to length(X)
  for do $j = 1$ to length(Y)
    $Match \leftarrow G(i-1,j-1) + S(x_i, y_j)$
    $Delete \leftarrow G(i-1,j) + d$
    $Insert \leftarrow G(i,j-1) + d$
    $G(i,j) \leftarrow \max(Match, Delete, Insert)$
  end for
end for
return G

Once the G matrix is calculated, the bottom right entry $G(n,m)$ gives the maximum score among all possible alignments. To find the alignment that actually gives this score, we start from $G(n,m)$, and compare the score with the three possible sources ("match", "insert" and "delete") to see from which it came. If "match", then x_i and y_j are aligned, if "delete", then x_i is aligned with a gap which means that x_i in the face track sequence should be deleted, and if "insert", then y_j is aligned with a gap which means that a gap should be inserted in the face track sequence. The process of finding the optimal alignment with the highest score is shown in Algorithm 2.

Algorithm 2. Algorithm for sequence alignment.

Input: Face track sequence X, name sequence Y, similarity matrix S, score matrix G
Output: Alignment result $AlignmentX$, $AlignmentY$

$AlignmentX \leftarrow$ " ", $AlignmentY \leftarrow$ " "
$i \leftarrow length(X)$, $j \leftarrow length(Y)$
while ($i > 0$ or $j > 0$) **do**
  if ($i > 0$ and $j > 0$ and $G(i,j) = G(i-1,j-1) + S(x_i, y_j)$) **then**

```
    AlignmentX ← x_i + AlignmentX
    AlignmentY ← y_j + AlignmentY
    i ← i − 1,   j ← j − 1
  else if (i > 0 and G(i, j) = G(i − 1, j) + d) then
    AlignmentX ← x_i + AlignmentX
    AlignmentY ← null + AlignmentY
    i ← i − 1
  else if (j > 0 and G(i, j) = G(i, j − 1) + d) then
    AlignmentX ← null + AlignmentX
    AlignmentY ← y_j + AlignmentY
    j ← j − 1
  end if
end while
return AlignmentX, AlignmentY
```

After sequence alignment, the faces can be annotated with the names aligned to them. For the rest faces which cannot find a name to align with, they are annotated with the name which are assigned to their face cluster. The name of the face cluster is determined by majority voting from the faces which have already gotten names in sequence alignment.

5. Experiments

The proposed approach in this paper is the extended version of our previous work [12]. In [12], we use the Levenshtein distance [13] measurement to find the optimal alignment between the face track sequence and the name sequence. The computational complexity is $O(K^L mn)$, where K is the number of the different names, L is the number of the face clusters, m is the length of the name sequence, and n is the length of the face track sequence. It is clear that when L increases, the method quickly becomes computationally intractable. In this paper, we use the Needleman–Wunsch algorithm to find the optimal alignment. The computational complexity is dramatically reduced to $O(mn)$. Besides comparing to our previous work [12] (abbreviated as "MinLeven"), we also make a comparison to another clustering based face annotation method called TVParser [11]. The evaluation is performed on two TV series, "Friends" and "I Love My Family"(abbreviated as "Family").

5.1. Data set configuration

"Friends" and "Family" are popular American and Chinese sitcoms, respectively. Our data set consists of six full episodes with three from "Friends" and three from "Family". Each episode usually has 20 min with around 6 main casts. In one episode of "Friends", there are about 20,000 face images which can be linked as about 400 face tracks, while in one episode of "Family" there are about 10,000 face images which can be linked as about 200 face tracks.

5.2. Quantitative analysis

Our method does not restrict the number of the face track clusters to the number of the main casts. In the experiments, we change the cluster number from 1 to 3 times of the main cast number. The purity of the clusters is expected to be increased when the cluster number increases. We want to investigate whether the final face naming results can be influenced by the cluster number. The results are average performances on "Friends" and "Family". Fig. 3 illustrates the performance of our approach.

In Fig. 3, the subfigures in the first column show the results on "Friends" and the subfigures in the second column show the results on "Family". In the first row, figure (a) and (b) show the average performance (the weighted recall, precision and F-score) of our method on face annotation. The weighted recall, precision and F-score are the weighted summation of the recall, precision and F-score of each character, respectively, where the weight is determined by the occurrence frequency of each character. Take the weighted F-score as the example, it is defined as follows:

$$F_w = \sum_i w_i \cdot \frac{2 \times Precision_i \times Recall_i}{Precision_i + Recall_i} \tag{5}$$

where w_i denotes the weight of the ith character which is determined by the occurrence frequency of his/her spoken lines in the script. For further details, figure (c) and (d) in the second row of Fig. 3 illustrate the F-score curves of the four most frequently appearing characters in each TV series.

It can be observed that most of the face tracks are named correctly. The average weighted F-scores on three episodes of "Friends" and "Family" are 0.77 and 0.78, respectively. The curves have certain fluctuation but do not clearly increase along with the increase of the cluster number, which demonstrates that our method is robust to the cluster number. It can be interpreted as follows: in our method, the similarity matrix between face clusters and names is the key factor which impacts the face naming result. Although the purity of the clusters increases along with the increase of the cluster number, the former large clusters tend to be split into smaller ones, resulting in the decrease of similarity between its temporal distribution and the true names, which make the similarity matrix less discriminative. On the other hand, the measurement based on the symmetric K–L divergence is robust to minor noises introduced by face clustering. Hence, when the cluster number is small and the purity of the clusters is relatively low, our method can still obtain satisfying annotation result.

In the third row, the comparison between TVParser [11], MinLeven method [12] and our current method is demonstrated in figure (e) and (f), where the F-score curves are shown. It can be seen that MinLeven method and our current method outperform TVParser. The current method also performs better than MinLeven method at the front area of the curves. TVParser achieves face–name association by matching the histograms of character occurrence frequency. Obviously, the frequency statistics contain much less information than the temporal sequence which includes more complex order constraints between the characters. For instance, two characters have the similar occurrence frequency, but their order along the time line are different. In this case, TVParser is more likely to confuse the two characters, but our two methods can differentiate them.

Based on the observation, we can find that the weighted F-score curve of TVParser reaches a peak and then decreases along with the increase of the cluster number. It means that the occurrence frequency histogram used in TVParser is more sensitive to the choice of cluster number. The performance of MinLeven method has a tendency to increase and becomes comparable to our current method when the cluster number increases. However, as we mentioned before, since MinLeven method have to enumerate all the possible alignment results, it quickly becomes computationally intractable when the cluster number is large. We compare the computation time cost on sequence alignment of our two methods, which is listed in Table 1. The experiments are conducted on Microsoft Windows Server 2003 X64 edition with Intel Xeon E5-2640@2.50 GHz. It is obvious that the average time cost of the current method is much less than MinLeven method. With the increase of the cluster number, the running time of MinLeven method increases much faster than the current one.

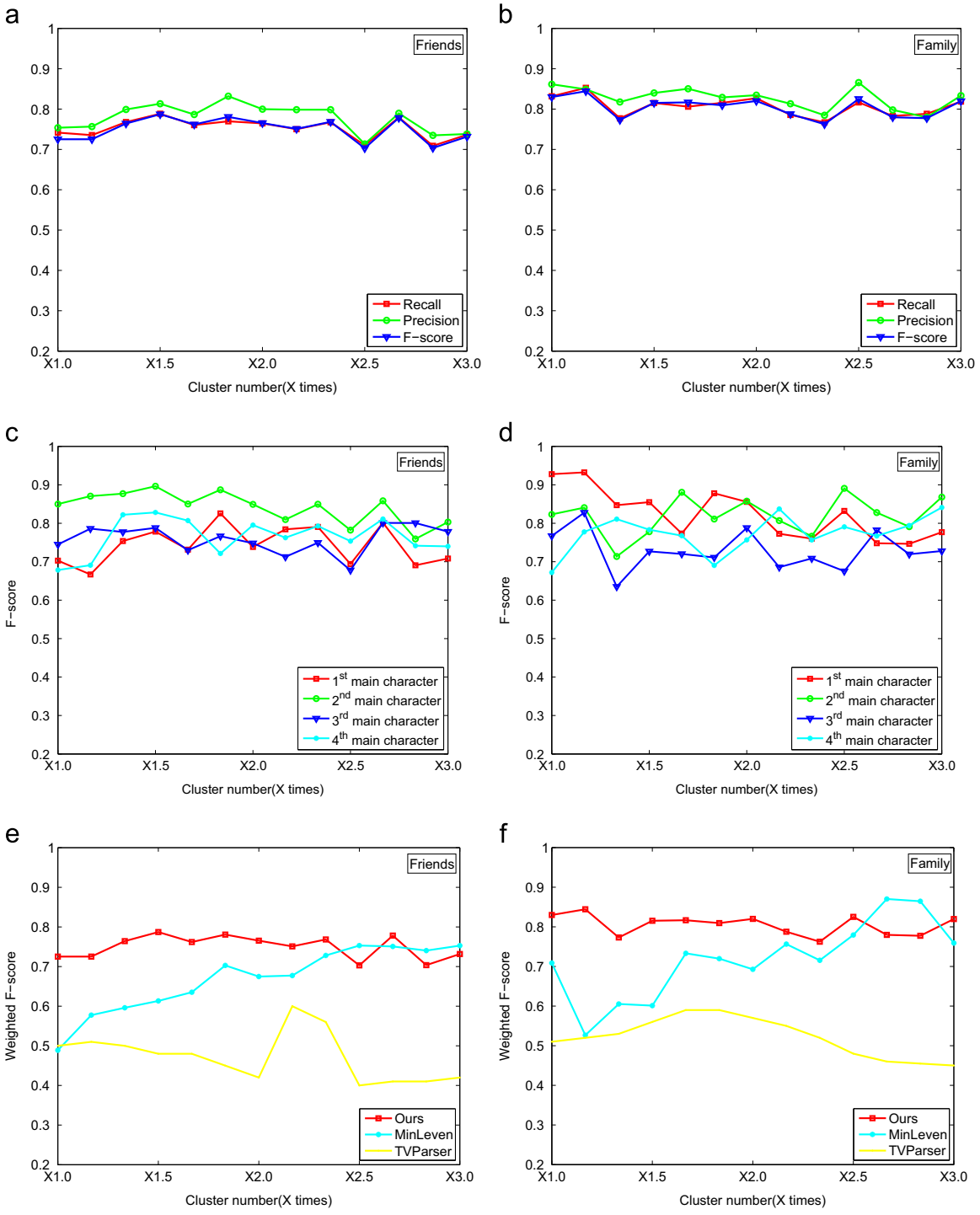


Fig. 3. Face annotation result in "Friends" (the first column) and "Family" (the second column). The first row (a,b) shows the performance of our method (recall, precision and weighted F-score). The second row (c,d) gives the detailed face annotation results of the top-4 main casts in the two TV series. The last row (e,f) presents the weighted F-score curves of three face annotation methods for comparison.

Table 1
Comparison of the running time (seconds) on sequence alignment.

Cluster number (× times)		× 1.0	× 1.5	× 2.0	× 2.5	× 3.0
Friends	MinLeven method [12]	11	16	1738	1599	55,342
	Current method	0.3416	0.3439	0.3561	0.3649	0.3673
Family	MinLeven method [12]	1	6	67	152	1930
	Current method	0.1572	0.1644	0.1635	0.1792	0.1869

Especially, when the cluster number is 3 times of the main casts, the running time on one episode of "Friends" is over 15 h, which is actually not practical.

6. Conclusions

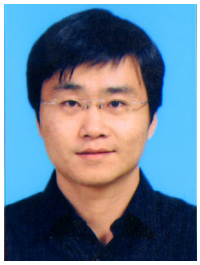
In this paper, we propose a new framework on face annotation in TV series videos based on the global sequence alignment between the name sequence from the script and the face track sequence from the video. We directly align the two heterogeneous sequences by introducing a similarity matrix measured by the symmetric K–L divergence and using a dynamic programming algorithm. Compared to our previous work which needs to enumerate all the possible alignment results, the current method significantly reduces the computational cost. The effectiveness and the efficiency of the work have been demonstrated in the experiments on two public TV series videos.

Acknowledgments

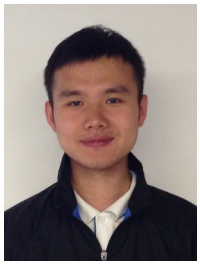
This work was supported in part by the 863 Program (2014AA015104), the National Natural Science Foundation of China (61202325, 61472422, 61303154, 61379100).

References

[1] M. Everingham, J. Sivic, A. Zisserman, "Hello! My name is… Buffy" automatic naming of characters in TV video, in: Proceedings of British Machine Vision Conference, 2006, pp. 889–908.
[2] J. Sivic, M. Everingham, A. Zisserman, "Who are you?"—learning person specific classifiers from video, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2009.
[3] M. Tapaswi, M. Bäuml, R. Stiefelhagen, "Knock! Knock! Knock! Who is it? Probabilistic person identification in tv-series, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2012.
[4] M. Bäuml, M. Tapaswi, R. Stiefelhagen, Semi-supervised learning with constraints for person identification in multimedia data, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2013.
[5] S.K. Pramod, C.V. Jawahar, A. Zisserman, Subtitle-free movie to script alignment, in: Proceedings of British Machine Vision Conference, 2009.
[6] B. Wu, Y. Zhang, B.-G. Hu, Q. Ji, Constrained clustering and its application to face clustering videos, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2013.
[7] Z. Lu, H.H.S. Ip, Constrained spectral clustering via exhaustive and efficient constraint propagation, in: Proceedings of European Conference on Computer Vision, 2010, pp. 1–14.
[8] R. Cinbis, J. Verbeek, C. Schmid, Unsupervised metric learning for face identification in TV video, in: Proceedings of International Conference on Computer Vision, 2011.
[9] Y.-F. Zhang, C. Xu, H. Lu, Y.-M. Huang, Character identification in feature-length films using global face–name matching, IEEE Trans. Multimed. 11 (7) (2009) 1276–1288.
[10] J. Sang, C. Xu, Robust face–name graph matching for movie character identification, IEEE Trans. Multimed. 14 (3) (2012) 586–596.
[11] C. Liang, C. Xu, J. Cheng, H. Lu, Tvparser: an automatic tv video parsing method, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2011, pp. 3377–3384.
[12] Z. Tang, Y. Zhang, H. Lu, Video face naming using global sequence alignment, in: Proceedings of International Conference on Image Processing, 2014.
[13] V. Levenshtein, Binary codes capable of correcting deletions, insertions and reversals, Sov. Phys. Dokl. 10 (1966) 707.
[14] S.B. Needleman, C.D. Wunsch, A general method applicable to the search for similarities in the amino acid sequence of two proteins, J. Mol. Biol. 48 (3) (1970) 443–453.
[15] Y. Sun, X. Wang, X. Tang, Deep convolutional network cascade for facial point detection, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2013, pp. 3476–3483.
[16] U.V. Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (4) (2007) 395–416.



Yifan Zhang received the B.E. degree in automation from Southeast University, in 2004 and the Ph.D. degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences, in 2010. Currently he is an Assistant Professor in the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. From 2011 to 2012, he was a Postdoctoral Research fellow in the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, New York. His research interests include multimedia content analysis, probabilistic graphical models and activity recognition.



Zhiqiang Tang received the B.E. degree from the Hebei University of Technology, Tianjin, China, in 2012. He is currently a Research Assistant at National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. His research interests include machine learning, algorithm and multimedia.

Chunjie Zhang received his Ph.D. degree in Pattern Recognition and Intelligent Systems from Institute of Automation, Chinese Academy of Sciences, China, in 2011. He received his B.E. degree from Nanjing University of Posts and Telecommunications, Jiangsu, China, in 2006. He worked as an Engineer in the Henan Electric Power Research Institute during 2011–2012. He is currently working as a Postdoc at School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing, China. Dr. Zhang's current research interests include image processing, machine learning, cross media content analysis, pattern recognition and computer vision.

Jing Liu received her B.E. degree in 2001 and M.E. degree in 2004 from Shandong University, and her Ph. D. degree from Institute of Automation, Chinese Academy of Sciences, in 2008. Currently she is an Associate Professor in National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. Her research interests include machine learning, image content analysis and classification, and multimedia information indexing and retrieval.

Hanqinq Lu (M'05–SM'06) received the Ph.D. degree in Huazhong University of Sciences and Technology, Wuhan, China, in 1992. Currently he is a Professor of Institute of Automation, Chinese Academy of Sciences. His research interests include image similarity measure, video analysis, object recognition and tracking. He published more than 100 papers in those areas.