Data organization in video surveillance systems using deep learning

A D Sokolova[1], A V Savchenko[1]

[1]National Research University Higher School of Economics, Bolshaya Pecherskaya str. 25/12, Nizhny Novgorod, Russia, 603155

Abstract. In this paper we propose to organize information in video surveillance systems by grouping the video tracks, which contain identical faces. Aggregation of the features of individual frames extracted using deep convolutional neural networks are used in order to obtain a descriptor of video track. The tracks with identical faces are grouped using the known face verification algorithms and clustering methods. We experimentally compare frame aggregation methods using the YouTubeFaces dataset and contemporary neural networks (VGGFace, VGGFace2, LightenedCNN). It is shown that the most accurate video-based face verification is achieved with the L2-normalization of average unnormalized features of individual frames of each video track. Finally, we demonstrate that the best video grouping is obtained by sequential and rank-order clustering methods.

1. Introduction

Nowadays, the automatic organization of visual information is attracting increasing attention due to the growth of the multimedia data volume. The multimedia data organization systems are required not only for a particular user who has an archive of photographs, but also for the field of public security, where video surveillance technologies are used [1, 2]. The limited functionality and the fact that the operator is physically unable to monitor the situation in real time with growth of the number of video cameras indicate the need to improve procedures and techniques, to increase the requirements for the training of operators. One approach is the use of intelligent video analytics systems to automatically process video streams.

Dramatic increase of size of collected media data leads to necessity of their grouping [3]. Existing solutions, e.g. Google Photos or Apple iPhoto, are designed to search, organize and display images of the person. However, they were not developed to process in real-time such large amount of data from video surveillance systems [4]. For example, they collect thousands of images (frames) every second [5, 6, 7]. Consequently, there is a challenge of ordering the visitors, whose faces were observed by a surveillance system [8].

Nowadays, the state-of-the-art results in image processing, object detection or feature extraction are obtained with deep convolutional neural networks (CNNs) [9, 10]. In this paper we use the clustering techniques in order to achieve automated organizing of video data where only one person is shown. As the quality of clustering mainly depends on the correctness of measuring the closeness of examined objects, in this paper we primarily focus on choosing the most appropriate representation of videos by aggregation of features obtained from each frame of video track.

The paper is organized as follows: in Section 2, we discuss the frame aggregation techniques. In Section 3, we present the proposed approach of video data organizing and our software prototype . In Section 4, the experimental results for the YouTubeFaces (YTF) dataset [9] are presented. In Section 5, the concluding comments and future plans are given.

2. Video frame aggregation techniques

The task of paper is to divide the input video sequence of $T>1$ frames into $M < T$ subsequent tracks *{X(m)}, m = 1,2,...,M* contained face images of one person and cluster similar tracks. Each *m*-th track is characterized by the indices of its start $t_1(m)$ and end frame $t_2(m)$. We denote the number of frames in the *m*-th track as $\Delta t(m)= t_2(m)- t_1(m)+1$.

In order to group tracks contained images of one person clustering methods were used [11, 12]. To utilize them it is necessary to extract face features in each frame [13, 14], aggregate features of separate frame in descriptor for whole track [15] and then compare these descriptors. The output of the CNN's last (bottleneck) layer of the facial image in the *t*-th frame is stored in the *D*-dimensional feature vector $\mathbf{x}(t)$. These features are usually matched with the Euclidean (L_2) metric $\rho(\mathbf{x}(t_1), \mathbf{x}(t_2))$ [16]. However, when the video sequences are grouped, it is required to compute the distance $\rho(X(m_1), X(m_1))$ between tracks (subsequences of frames) $X(m_1)$ and $X(m_2)$. The most obvious way to define this distance is the computation of the mean pairwise distances between all frames:

$$\rho\left(X(m_1), X(m_2)\right)=\frac{1}{\Delta t(m_1)\Delta t(m_2)}\sum_{t=t_1(m_1)}^{t_2(m_1)}\sum_{t'=t_1(m_2)}^{t_2(m_2)}\rho\left(\mathbf{x}(t),\mathbf{x}(t')\right) \tag{1}$$

However, the run-time complexity is rather high due to the pair-wise matching of all frames in these tracks causing the computation of $\Delta t(m_1)\Delta t(m_2)$ distances between high-dimensional features. Therefore, we used the following methods to match single representations of the whole tracks.

1. The distance between their medoids:

$$\rho\left(X(m_1), X(m_2)\right)=\rho\left(\mathbf{x}^*(m_1),\mathbf{x}^*(m_2)\right),$$
$$\mathbf{x}^*(m_i)=\underset{\mathbf{x}(t),t\in[t_1(m_i),t_2(m_i)]}{\operatorname{argmin}}\sum_{t'=t_1(m_i)}^{t_2(m_i)}\rho\left(\mathbf{x}(t),\mathbf{x}(t')\right) \tag{2}$$

2. Average features vectors of each track are matched:

$$\rho\left(X(m_1), X(m_2)\right)=\rho\left(\bar{\mathbf{x}}(m_1),\bar{\mathbf{x}}(m_2)\right),$$
$$\bar{\mathbf{x}}(m_i)=\frac{1}{\Delta t(m_i)}\sum_{t'=t_1(m_i)}^{t_2(m_i)}\mathbf{x}(t) \tag{3}$$

3. Comparison of the median features $\mathbf{x}'(m_i)$ of each track:

$$\rho\left(X(m_1), X(m_2)\right)=\rho\left(\mathbf{x}'(m_1),\mathbf{x}'(m_2)\right). \tag{4}$$

4. Learnable pooling of video features of the *m*-th track using the neural aggregation network [17], which includes special attention blocks to process features of all frames. The goal of this block is to learn the *D*-dimensional vector $\mathbf{q}$, which is used to assign each *t*-th frame with its weight $a(t)$ using the softmax function:

$$a(t)=\frac{\exp(\mathbf{q}^T\mathbf{x}(t))}{\sum_{t'=t_1(m)}^{t_2(m)}\exp(\mathbf{q}^T\mathbf{x}(t'))} \tag{5}$$

The final representation of the *m*-th track is computed as the weighted average of all frames with weights (5):

$$\mathbf{r}(m)=\sum_{t=t_1(m)}^{t_2(m)}a(t)\mathbf{x}(t) \tag{6}$$

In order to improve the quality of such learnable pooling it is significant to use two sequential attention blocks [17]. Let $\mathbf{q}^{(0)}$ be the first block weights and $\mathbf{r}^{(0)}$ be the first aggregating features using $\mathbf{q}^{(0)}$ weights (5), (6). The weights in the sequential block are computed as follows:

$$\mathbf{q}^{(1)} = \tanh(W\mathbf{r}^{(0)} + b), \quad (7)$$

where W and b are the learnable weight matrix and bias vector of the neurons respectively. The feature vector $\mathbf{r}^{(1)}$ generated by $\mathbf{q}^{(1)}$ using (6) will be the final descriptor of the m-th frame.

In order to make the video features more resistant to the conditions of observation (camera resolution, illumination, etc.) usually normalization in the Euclidean metric is applied [10]. Usually, the preliminary normalization of the features of each frame is performed. However, in this paper we also analyze the normalization of aggregated video features [18].

3. Proposed video data organizing system

We implemented a special software prototype using PyCharm from JetBrains (Python 3.6 language), OpenCV [19], Caffe and TensorFlow libraries. The data flow in our system is presented in Figure 1.

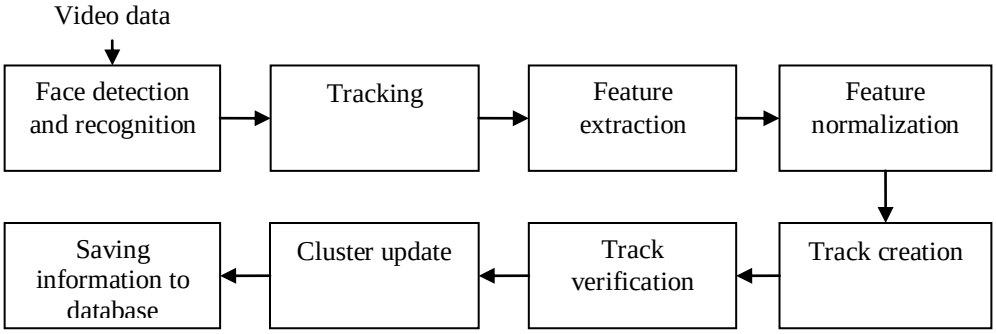

Figure 1. Dataflow of data organization in video surveillance systems.

At first, the faces are detected in each frame with the TensorFlow Models [20]. This repository contains different preliminary trained models of neural networks and provides interface for object detection (TensorFlow Object Detection API). In particular, faces are detected with the MobileNet SSD trained on the WiderFace dataset [21]. Further tracking of highlighted faces is processed but face detection is repeated periodically in order to: 1) precise tracking results; 2) obtain new faces and 3) mark disappeared faces. After that we extract features using particular CNN and normalize the features vector. Then consecutive frames of one person are united into one track (homogeneous segment). On the final step (Fig.1) subsequent clustering is processed: features vector of the last track is matched with the features of previously detected clusters. If the distance to the nearest cluster does not exceed a certain threshold, this track is added to the cluster and the information about the last is updated. The resulted set of selected clusters of homogeneous tracks is saved into NoSQL database Cassandra.

The architecture of the proposed software is shown in Fig. 2.

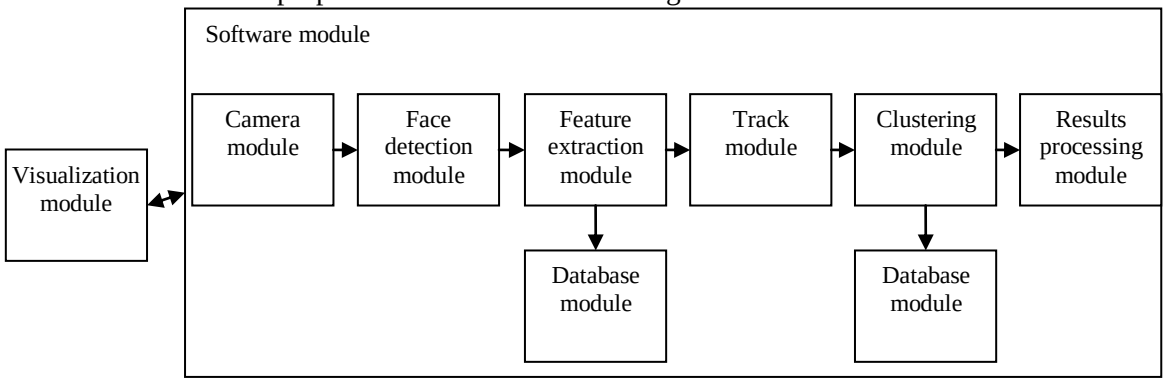

Figure 2. The architecture of the proposed software.

Using the cross-platform Qt framework for software development a graphical user interface was created (Fig. 3).

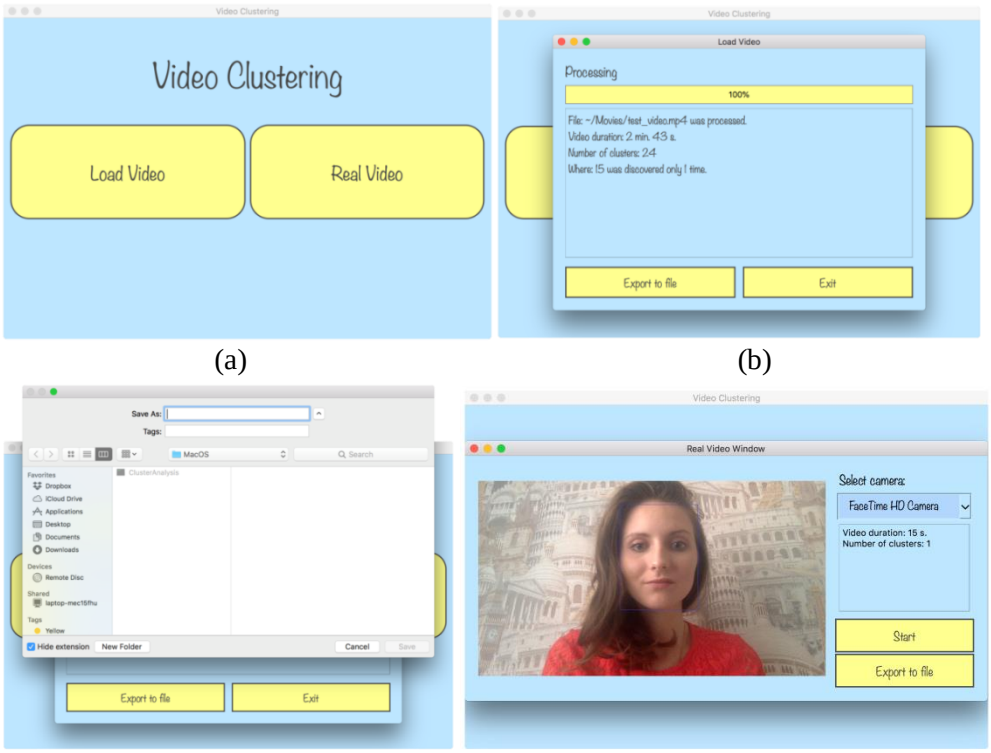

Figure 3. Results of working system (*a* – the main window, *b* – loading video processing and results, *c* – export data to file, *d* – real video processing and results).

4. Experimental results

In this section we describe the experimental results of our pipeline (Fig. 1). To extract features we used the Caffe framework [9] and three publicly available CNNs suitable for face recognition, namely, the VggNet [22], Lightened CNN (version C) [10], VggFace2 [23]. The VggNet extracts D = 4096 features vector in the output of "fc7" layer from 224x224 RGB images. The Lightened CNN extracts D = 256 features vector ("eltwise fc2" layer) is computed from 128x128 grayscale image. The VggFace2 is the ResNet50 model, which extracts D = 2048 features vector from "pool5_7x7_s1". Their advantages are high velocity of image processing and high accuracy of detection.

In this paper we analysed two types of distance between frames[24, 25]: traditional L2 (Euclidiean) metric and the Student criterion (t-test):

$$t = \frac{\rho(X(m_1), X(m_2))}{\sqrt{\frac{D(m_1)}{\Delta t(m_1)} + \frac{D(m_2)}{\Delta t(m_2)}}} \quad (8)$$

Here we used the YTF dataset [26], which contains 3,425 videos of 1,595 different people. The shortest track duration is 48 frames, the longest track contains 6,070 frames, and the average length of a video clip is 181.3. We have calculated following indexes: AUC (Area under curve) and FRR (False Reject Rate) for fixed FAR (False Accept Rate) = 1%. Weights in attention modules (5)-(7) were trained on 1895 videos of 500 subjects from the IJB-A dataset.

The AUC (area under curve) and the false reject rate (FRR) for 1% false accept rate (FAR) for Lightened CNN, VggNet and VggFace2 features are shown in Table 1, Table 2 and Table 3, respectively. Average face detection time for one frame is 60 ms at MacBook Pro laptop.

According to these tables, one can clearly see that the proper normalization plays a significant role. The most efficient algorithm is computation of average vector of normalized features. AUC of average vector search is more by 10-12% than AUC of medoid (2) comparison and by 14-16% than AUC of aggregation module (5)-(7) training.

Table 1. Video-based face verification results, Lightened CNN.

Measure	Distance between tracks	AUC (%)	FRR@FAR=1%
Distance (1)	L_2	90.7±0.6	77.0±8.4
L_2-norm->Distance (1)	L_2	98.2±0.4	14.1±3.6
Medoids (2)	L_2	89.7±0.6	80.6±6.4
	t-test (8)	84.7±0.7	72.9±7.8
L_2-norm (2) medoids	L_2	97.2±0.6	19.1±4.3
	t-test (8)	88.8±0.6	54.1±5.9
Attributes averaging(3)	L_2	91.3±1.3	71.8±10.0
	t-test (8)	91.8±1.4	72.3±11.5
L_2-norm attributes averaging (3)	L_2	97.7±0.5	21.4±6.4
	t-test (8)	96.8±0.5	37.2±7.6
L_2-norm of average attributes vector (3)	L_2	98.3±0.7	12.4±3.1
	t-test (8)	97.6±0.5	12.5±3.1
L_2-norm of median (4)	L_2	96.7±0.6	22.3±7.2
	t-test (8)	94.4±0.5	37.0±7.5
Attention module (5)-(7)	L_2	87.6±0.9	65.3±6.5
	t-test (8)	89.3±1.2	64.8±7.0
L_2-norm of attention module (5)-(7)	L_2	90.2±0.7	66.7±5.4
	t-test (8)	90.4±0.8	54.5±9.3

Table 2. Video-based face verification results, VggNet.

Measure	Distance between tracks	AUC (%)	FRR@FAR=1%
Distance (1)	L_2	83.3±0.8	85.8±9.0
L_2-norm->Distance (1)	L_2	97.9±0.6	23.2±6.3
Medoids (2)	L_2	85.7±1.8	86.0±8.4
	t-test (8)	80.8±1.2	83.9±7.7
L_2-norm (2) medoids	L_2	93.5±1.1	25.4±6.2
	t-test (8)	85.2±0.7	69.9±7.9
Attributes averaging(3)	L_2	89.1±0.9	79.7±7.8
	t-test (8)	87.4±1.2	81.2±5.8
L_2-norm attributes averaging (3)	L_2	97.2±0.6	54.4±6.3
	t-test (8)	96.3±0.7	76.9±6.8
L_2-norm of average attributes vector (3)	L_2	98.1±1.0	19.4±5.9
	t-test (8)	97.7±0.6	25.3±7.8
L_2-norm of median (4)	L_2	96.2±0.7	32.4±6.5
	t-test (8)	94.8±0.7	41.1±7.3
Attention module (5)-(7)	L_2	87.8±0.4	54.3±5.8
	t-test (8)	89.7±0.8	52.7±6.0
L_2-norm of attention module (5)-(7)	L_2	90.1±0.7	46.9±5.9
	t-test (8)	90.0±0.6	44.5±6.4

Table 3. Video-based face verification results, VggFace2.

Measure	Distance between tracks	AUC (%)	FRR@FAR=1%
Distance (1)	L_2	98.5±0.5	94.4±3.3
L_2-norm->Distance (1)	L_2	99.0±0.6	94.4±3.2
Medoids (2)	L_2	96.0±0.4	94.9±4.5
	t-test (8)	87.7±0.6	94.6±3.5
L_2-norm (2) medoids	L_2	98.4±0.7	94.9±3.6
	t-test (8)	86.2±0.7	94.6±4.1
Attributes averaging(3)	L_2	98.8±0.6	93.0±4.4
	t-test (8)	98.0±0.5	93.1±3.8
L_2-norm attributes averaging (3)	L_2	99.1±0.6	92.1±4.0
	t-test (8)	95.8±0.8	92.3±5.5
L_2-norm of average attributes vector (3)	L_2	98.5±0.5	92.9±6.3

L_2-norm of median (4)	t-test (8)	94.0±0.6	93.1±4.8
	L_2	94.9±0.7	94.6±5.6
Attention module (5)-(7)	t-test (8)	98.5±0.4	94.7±3.0
	L_2	88.4±0.6	95.7±3.4
L_2-norm of attention module (5)-(7)	t-test (8)	89.9±0.8	92.6±6.2
	L_2	91.7±0.7	93.4±4.5
	t-test (8)	92.2±0.7	94.4±4.5

Also we implemented sequential clustering where threshold for resulting clusters is set by fixing the FAR value. In addition, we examined the clustering algorithm [27] from the DominantSet library [28] and the Rank-Order hierarchical clustering [29]. The results are presented in Table 4.

Table 4. Clustering results.

	Lightened CNN		**VggNet**		**VggFace2**	
	Total quantity of clusters	**Quantity of incorrect clusters**	**Total quantity of clusters**	**Quantity of incorrect clusters**	**Total quantity of clusters**	**Quantity of incorrect clusters**
Sequential clustering, FAR=1%	2492	35	2634	44	2102	23
Sequential clustering, FAR=10%	2147	162	2195	171	1956	121
DominantSet	1836	180	1902	225	1801	177
Rank-Order	2105	36	2213	42	2007	31

Total quantity of clusters is bigger than quantity of different people from YTF dataset (1,595) because different videos with one person could be mentioned in different clusters. Moreover, average processing time of YTF tracks for hierarchical algorithm is 8 minutes, while algorithm DominantSet utilized for video tracks grouping consumed more than 20 minutes. The most effective results were demonstrated by Rank-Order clustering algorithm. As usual, the most accurate results are obtained for the VggFace2 facial features.

5. Conclusion

In this paper we solved the problem of video subsequences clustering for video surveillance systems. In particular, we focused on calculating the degree of proximity of video tracks using the aggregation of features vectors extracted by deep CNNs. Experimental study demonstrated that the features vectors averaging of all frames and subsequent normalization lead to the highest accuracy of video face verification. In the future work we plan to analyze other clustering algorithms deeper in order to achieve low calculation complexity and high accuracy of data processing.

6. References

[1] Chellappa R, Du M, Turaga P and Zhou S K 2011 Face Tracking and Recognition in Video *Handbook of Face Recognition* 323-351
[2] Shan C 2016 Face Recognition and Retrieval in Video *Video Search and Mining, Studies in Computational Intelligence* **287** 235-260
[3] Savchenko A V 2016 Search Techniques in Intelligent Classification Systems *Springer International Publishing*
[4] Savchenko A V and Belova N S 2018 Unconstrained Face Identification Using Maximum Likelihood of Distances Between Deep Off-the-shelf Features *Systems with Applications* **108** 170-182

[5] Chen J C, Ranjan R, Kumar A, Chen C H, Patel V M and Chellappa R 2015 An end-to-end system for unconstrained face verification with deep convolutional neural networks *IEEE International Conference on Computer Vision Workshops* 118-126
[6] Li H, Hua G, Shen X, Lin Z and Brandt J 2014 Eigen-PEP for video face recognition *Asian Conference on Computer Vision, LNCS* **9005** 17-33
[7] Savchenko A V 2017 Deep neural networks and maximum likelihood search for approximate nearest neighbor in video-based image recognition *Optical Memory and Neural Networks (Information Optics)* **26(2)** 129-136
[8] Sokolova A D, Kharchevnikova A S and Savchenko A V 2017 Organizing Multimedia Data in Video Surveillance Systems Based on Face Verification with Convolutional Neural Networks *Lecture Notes in Computer Science* **10716** 213-220
[9] Jia Y et al 2015 Caffe: Convolutional architecture for fast feature embedding *Proceedings of the 22nd ACM international conference on Multimedia ACM* 675-678
[10] Wu X, He R and Sun Z 2015 A Lightened CNN for Deep Face Representation *Preprint arXiv*: *1511.02683*
[11] Kaufman L and Rousseeuw P J 2009 *Finding groups in data: an introduction to cluster analysis* (John Wiley & Sons)
[12] Savchenko A V 2017 Clustering and maximum likelihood search for efficient statistical classification with medium-sized database *Optimization Letters* **11(2)** 329-341
[13] Savchenko A V 2018 Trigonometric series in orthogonal expansions for density estimates of deep image features *Computer Optics* **42(1)** 149-158 DOI: 10.18287/2412-6179-2018-42-1-149-158
[14] Savchenko A V 2017 Maximum-likelihood dissimilarities in image recognition with deep neural networks *Computer Optics* **41(3)** 422-430 DOI: 10.18287/2412-6179-2017-41-3-422-430
[15] Nikitin M Yu, Konushin V S and Konushin A S 2017 Neural network model for video-based face recognition with frames quality assessment *Computer Optics* **41(5)** 732-742 DOI: 10.18287/2412-6179-2017-41-5-732-742
[16] Goodfellow I, Bengio Y and Courville A I 2016 *Deep learning* (MIT press)
[17] Yang J 2017 Neural aggregation network for video face recognition *arXiv Preprint*
[18] Savchenko A V and Belova N S 2015 Statistical testing of segment homogeneity in classification of piecewise- regular objects *International Journal of Applied Mathematics and Computer Science* **25(4)** 915-925
[19] *OpenCV* (Access mode: http://opencv.org/)
[20] *TensorFlow API* (Access mode: http://github.com/tensorflow/models/tree/master/research/object_detection/)
[21] *Wider Face: A face detection benchmark* (Access mode: http://mmlab.ie.cuhk.edu.hk/projects/WIDERFace/)
[22] Parkhi O M, Vedaldi A and Zisserman A 2015 Deep face recognition *Proceedings of the British Machine Vision* 6-17
[23] Cao Q, Shen L, Xie W, Parkhi O M and Zisserman A 2017 VGGFace2: A dataset for recognising faces across pose and age *Preprint arXiv: 1710.08092*
[24] Nemirovskiy V B, Stoyanov A K and Goremykina D S 2016 Face recognition based on the proximity measure clustering *Computer Optics* **40(5)** 740-745 DOI: 10.18287/2412-6179-2016-40-5-740-745
[25] Nemirovskiy V B and Stoyanov A K 2017 Clustering face images *Computer Optics* **41(1)** 59-66 DOI: 10.18287/2412-6179-2017-41-1-59-66
[26] Wolf L, Hassner T and Maoz I 2011 Face recognition in unconstrained videos with matched background similarity *IEEE International Conference on Computer Vision and Pattern Recognition* 529-534
[27] Pelilo M and Pavan M 2007 Dominant sets and pairwise clustering *IEEE Transactions on Pattern Analysis and Machine intelligence* **29(1)** 167-172
[28] *Dominant Set Library* (Access mode: https://github.com/xwasco/DominantSetLibrary)

[29] Zhu C, Wen F and Sun J 2011 A rank-order distance based clustering algorithm for face tagging *IEEE International Conference on Computer Vision and Pattern Recognition* 481-488

Acknowledgements
The work was conducted at Laboratory of Algorithms and Technologies for Network Analysis, National Research University Higher School of Economics and supported by Russian Federation President grant MD-306.2017.9.