Facial Peculiarity Retrieval via Deep Neural Networks Fusion

Peiqin Li, Jianbin Xie, Zhen Li, Tong Liu, Wei Yan

*School of Electronic Science, National University of Defense Technology,
Changsha, Hunan, P. R. China. 410073,*

E-mail: lipeiqin_nudt@163.com

Received 26 July 2016

Accepted 15 September 2017

Abstract

Face retrieval is becoming increasingly useful and important for security maintenance operations. In actual applications, face retrieval is usually influenced by some changeable site conditions, such as various postures, expressions, camera angles, illuminations, and so on. In this paper, facial peculiar features are extracted and classified by dynamically integrated deep neural networks (DNNs), in order to enhance the adaptability in actual conditions. Firstly, eight kinds of facial components are detected and located by clustering analysis and Active Shape Model (ASM). Secondly, certain peculiar patterns are defined for each kind of facial component, and eight specialized DNNs are designed to extract features and classify components. Thirdly, the similarity between faces is calculated by dynamically integrating the results of each DNN. Comparative experiments on standard image sets and wild image sets demonstrate that our algorithm outperforms global feature models in retrieval accuracy. Our algorithm is particularly suitable for practical application with regard to natural real videos and images.

Keywords: face retrieval; clustering analysis; ASM; deep learning; DNN.

1. Introduction

Video monitoring systems have been employed in the field of security worldwide, and face retrieval has become one of the well-studied problems in computer vision. Finding coincident faces from captured videos or images remains a challenge and requires development. Although videos can provide more information than a single image¹, and several existing methods have performed impressively on face retrieval^{2, 3}, these methods are primarily developed and tested by using either strictly controlled footage or high-quality video images. Faces in these materials are often collaborative and shot under simple lighting and viewing conditions, and videos or images are screened and stored in high-quality format.

Videos in practical repositories are different, because they are used to record real scenes such as streets, public squares, bus or train stations, airports, etc. Such videos are typically characterized by changeful postures and complex lighting conditions, and are often corrupted by motion blur. Bandwidth and storage

limitations may result in compression artifacts that make face retrieval even more difficult.

Some effective face recognition algorithms have been studied by many groups such as Google, Face++, etc. We pay attention apart from others to face retrieval from actual constrained monitoring videos or images, and we propose a novel framework at the base of facial peculiarity and deep learning, as shown in Fig. 1.

The contributions of this study are summarized as follows:

- (1) Clustering analysis is implemented to reduce the target region, so that the face detection and facial component location can be accelerated.
- (2) The peculiar features of components are distinct and stronger than those of entire faces.
- (3) A robust model by DNN is used to classify facial components. This model is more effective in extracting high-level features than traditional neural networks with fewer layers.
- (4) A dynamic method is introduced to synthesize multiple DNNs. This method can effectively reflect the peculiar intensities of different components.

Fig. 1 Framework of our algorithm. Eight kinds of components are chosen for the operation: left brow, right brow, left eye, right eye, nose, mouth, beard, and visible scars or birthmarks. The first six are detected and located by ASM from videos or images, and the last two are obtained by checking correlative areas of faces. For videos of known categories, the obtained components are classified accordingly and used to train DNN models. The template face and unknown faces are handled separately, and the results are dynamically integrated to calculate the complete similarity for the retrieval.

2. Related work

Previous research focused on searching face targets obtained from different modalities. For example, an original face image is converted into a corresponding sketch image, after which recognition is conducted by sketches^{4, 5}. Such a method can simplify the facial features, but the sketch extraction may be influenced easily by illumination. Certain handcrafted descriptors, such as local binary pattern (LBP), scale invariant feature transform (SIFT), and histogram of oriented gradients (HOG)^{6, 7, 8} have the effect of comparing faces, but they generally consider the features of the whole face and are easily disturbed by postures and illuminations. Boosting methods are used to detect facial key points^{9, 10}, because such methods can accurately locate the face. However, many such studies are used as bases of controlled image sets in laboratories, such as MultiPIE and FERET benchmarks^{11, 12}, in which

the image-forming condition is strictly controlled. The condition is considerably more complex in fact.

For faces with posture and lighting variation, 3D models are used to revise an off-axis image to a frontal image and to standardize lighting suitable for comparison^{13, 14, and 15}. Although these methods may have reliable effects, they are time consuming and typically require additional and special imaging equipment.

Deep learning, a class of machine learning techniques mainly developed since 2006, has been applied for face classification. Deep learning generates numerous stages of nonlinear information processing in hierarchical architectures, which are employed in feature learning and pattern classification¹⁶. Typical models contain deep belief network, restricted Boltzmann machine, deep Boltzmann machine, DNN, and related unsupervised learning algorithms such as auto encoders¹⁷ and sparse coding¹⁸. These methods are used for higher-level feature representations and classification^{19, 20}. In recent years, some famous companies and research teams have used DNN for face detection or recognition, such as Facebook's DeepFace²¹, Yahoo's Deep Dense Face Detector²², Google's FaceNet²³, and so on. Generally, deep learning methods turn a global face image as input data, and face analysis is accomplished internally. On the one hand, this process shows the intelligent advantage of deep learning, on the other hand, feature extraction of a whole face increases the complexity of networks. Retrieval accuracy may be influenced, particularly when the videos and images are obtained from actual applications.

In this study, we propose a robust and useful algorithm, which takes advantage of deep learning with peculiarities of facial components.

3. Facial component location

For facial component location, Active Shape Model (ASM) is a classic algorithm introduced by Cootes²⁴ and improved by other researchers over the past few years. ASM is used to automatically locate landmark points that define the shape of any statistically modeled object from an image. When modeling faces, the landmark points lie along the shape boundaries of facial components such as eyebrows, eyes, nose, lips, and so on. Searching the best candidate feature points by traditional ASM needs long time. In this paper we propose improvement approaches to increase the rate.

To strengthen the adaptability of global variations, some pre-treatments are needed, such as adjustment and normalization of global brightness. And then considering face usually has conformable local gray value, while other regions have diverse gray value, such as textures and graphic patterns on clothes, we should take advantage of the gray-similarity to detect possible areas. Clustering analysis can achieve the pattern recognition based on the similarities to judge automatically²⁵, so we actualize the clustering analysis by K-Means, in order to reduce the object regions of ASM. The process is expounded as follows:

Algorithm 1: Clustering analysis based on K-Means

Input : Sample pixels x of image containing faces, the number of whole patterns k .

Output : The final clustering classes $\{S_1, S_2, \dots, S_k\}$.

```

1: Initialization: Randomly choose centers of
   clustering as  $C_1^{m1}, C_2^{m2}, C_3^{m3}, \dots, C_k^{mk}$ , where
    $m1, m2, m3, \dots, mk$  separately means the number
   of iteration operation.
2: while  $C_i^{r+1} \neq C_i^r, (i=1, 2, 3, \dots, k)$  do
3:   for  $i=1:k$ 
4:     if  $d(x, C_i^r) = \min(d(x, C_j^r), j=1, 2, 3, \dots, k)$ 
5:        $S_i^r \leftarrow x$ 
6:        $C_i^{r+1} = \frac{1}{n_i^r} \sum_{x \in C_i^r} x, (i=1, 2, 3, \dots, k)$ 
7:        $n_i^r$  is the number of samples belonging
8:       to the  $S_i^r$ .
9:     end if
10:  end for
11: end while
12: Cluster samples  $x$  from  $I$  into  $k$  patterns.

```

By these treatments, the searching range of ASM can be reduced, and the whole process can achieve higher speed. The location of facial component is annotated on training image set. Supposing that there are n face images in the training set and each face has m landmark points, the shapes can be represented by vectors

$$X^i = (x_0^i, y_0^i, x_1^i, y_1^i, \dots, x_{m-1}^i, y_{m-1}^i)^T, i = 0, 1, \dots, n-1 \quad (1)$$

Here x_j^i and y_j^i are the coordinates of the j th landmark in i th face image.

By aligning faces in set X , we can get the new set

$$\hat{X}^i = (\hat{x}_0^i, \hat{y}_0^i, \hat{x}_1^i, \hat{y}_1^i, \dots, \hat{x}_{m-1}^i, \hat{y}_{m-1}^i)^T, i = 0, 1, \dots, n-1 \quad (2)$$

And the average template of face is

$$\bar{X} = \frac{1}{n} \sum_{i=0}^{n-1} \hat{X}^i \quad (3)$$

The deviation between samples and average template can be calculated as

$$dX^i = \hat{X}^i - \bar{X} \quad (4)$$

And the covariance matrix is

$$S = \frac{\sum_{i=0}^{n-1} (dX^i)(dX^i)^T}{n} \quad (5)$$

We assume the nonzero eigenvalue vector and eigenvector are λ_i and p_i , then a face shape can be represented as

$$X = \bar{X} + P_t B_t \quad (6)$$

Here $P_t = (p_1, p_2, \dots, p_t)$ is a matrix of the first t eigenvectors of the covariance matrix, and $B_t = (b_1, b_2, \dots, b_t)$ is a vector that indicates the variation from X to $\bar{X}$ in the direction of $p_1, p_2, \dots, p_t$. Across the training set, the variance of the i th parameter vector p_i is given by the corresponding eigenvalue vector λ_i . To ensure the generated shape is

Fig. 2 Examples of facial component location. Red points mean the landmark points of facial components. After facial component location, it is necessary to increase the consistency of images. This task is conducted by aligning, which includes translocation, zooming, and rotation.

similar to those in the training set, b_i should meet the constraint condition as $|b_i| \leq 3\sqrt{\lambda_i}$, which is a commonly-used configuration in the ASM domain. By steps above, face shapes in videos and images can be searched and extracted, and then facial components can be located with the help of landmark points. Several examples are presented in Fig. 2.

4. Face retrieval by dynamically integrated DNN

DNN is a deep learning model that functions as both feature extractor and classifier. For feature extraction, it maps specific pixels from an input image into a general and hierarchical feature vector. The feature vector can be classified by several fully connected layers^{26, 27}. In contrast to traditional methods, DNN has higher artificial intelligence, that is, certain functions and parameters can be optimized inside automatically but

not artificially by training, and better effects and higher efficiency can be achieved. In this paper, features of separate facial component are extracted by traditional DNN at first, and then, an innovative fusion structure is proposed to dynamically integrate the results of each DNN.

Generally, DNN has a succession of layers, including an input layer, an output layer, and several hidden layers with multiple units. The input layer is the image data, the output layer is the result, and each hidden layer generates mapping vectors. Every layer directly links only to the one behind it except the output layer. The data in each layer is transformed by convolutional function, which is related to the activeness of the corresponding units in its layer. According to the training process, the adjustable parameters of DNN are jointly optimized by minimizing misclassification error. The structure of DNN is shown in Fig. 3.

Fig. 3 The structure of single DNN

We suppose that the unit j in layer n is Y_j^n , then layer n performs a 2D convolution of its M^{n-1} input maps with a filter of size $w_j^n \times h_j^n$. The resulting activations of the M^n output maps are given by the sum of the M^{n-1} convolutional responses, which pass through the following nonlinear activation function:

$$Y_j^n = \text{sigm}\left(\sum_{i=1}^{M^{n-1}} Y_i^{n-1} W_{ij}^n + b_j^n\right) \quad (7)$$

Here n is the layer index; Y is a map of size $M_w \times M_h$, W_{ij} is the weight vector between layer n and layer $n+1$, b_j^n is the bias of output map j , and sigm is the sigmoid squashing function, as follows:

$$\text{sigm}(a) = \frac{1}{1 + e^{-a}} \quad (8)$$

In layer $n-1$, the output map Y^{n-1} with the size of $M_w^{n-1} \times M_h^{n-1}$ becomes the input map of layer n .

After convolution with the size of $w_j^n \times h_j^n$, the output map Y^n has a size of:

$$\begin{aligned} M_w^n &= M_w^{n-1} - w_j^n + 1 \\ M_h^n &= M_h^{n-1} - h_j^n + 1 \end{aligned} \quad (9)$$

In the last layer $l+1$, the final output is:

$$Y^{l+1} = f\left(\sum_{i=1}^{M^{n-1}} Y_i^l W_i^{l+1} + b^{l+1}\right) \quad (10)$$

The activation function f depends on the supervised task that the network must achieve. Typically, it is the identity function for a regression problem and is named softmax function, as follows:

$$f_j(a) = \text{soft max}_j(a) = \frac{e^{a_j}}{\sum_{i=1}^L e^{a_i}} \quad (11)$$

In this formula, L represents the whole number of patterns to be classified.

By training on a known image set, the final output layer is down sampled to one pixel or a one-dimensional feature vector. The training process can be conducted as follows:

Algorithm 2: Training of DNN

Input : The known image sets $\{I, Y\}$ for training, the training modificatory factor $\mathcal{E}_t$, and correct ratio C_t of classification. Where I is the image data, and Y is corresponding classification.

Output : The weights and biases of layers in DNN.

- 1: **Initialization:** Set the weight vector W_{ij}^n randomly with the distribution of $U(a^{-0.5}, a^{-0.5})$, and set $b_j^n = 0$, where $a = \max(\|Y_j^{n-1}\|, \|Y_j^n\|)$.
- 2: **while** correct ratio of classification $\leq C_t$ **do**
- 3: **for** $i = 1:l$
- 4: $h^0(x_i) \leftarrow x_i$
- 5: **for** $j \in \{1, \dots, i-1\}$
- 6: $a^j(x_i) = b^j + W^j h^{j-1}(x_i)$
- 7: $h^j(x_i) = \text{sigm}(a^j(x_i))$
- 8: **end for**
- 9: $o(x_i) = h^{l+1}(x_i) = \text{softmax}(a^{l+1}(x_i))$
- 10: **end for**
- 11: compute current correct ratio of classification
- 12: $b^{l+1} \leftarrow b^{l+1} + \mathcal{E}_{\text{fine-tune}} \frac{\partial \log o_{y_i}(x_i)}{\partial a^{l+1}(x_i)}$
- 13: $W^{l+1} \leftarrow W^{l+1} + \mathcal{E}_{\text{fine-tune}} \frac{\partial \log o_{y_i}(x_i)}{\partial a^{l+1}(x_i)} h^l(x_i)^T$
- 14: **end while**
- 15: Generate the DNN with weights and biases.

For face retrieval, the images of eight kinds of facial components are sent to corresponding DNNs and later provide eight classification results of the components. The similarity between the detected face and template face can be calculated upon synthesis of single results. Therefore, a key concern is how to optimize the combination of outputs from various DNNs. The common operation is simply averaging the outputs. In our algorithm, all DNNs with dynamic weights are integrated, and we take the different significance of every facial component into account. The process is presented in Fig. 4.

For component C_i , the DNN is N_i , and the component is classified to pattern p , with the weight U_i . For pattern p , the final standard mapping vector is V_i^p , and the final mapping vector of input facial component is V_i^t . The distance between the classification result and the standard pattern is:

$$D_i = \|V_i^t - V_i^p\| \quad (12)$$

The whole difference of the face is:

$$J_m(U) = \sum_{i=1}^K U_i^m D_i \quad (13)$$

K is the whole pattern kind valued as 8 in this study. In this paper, every facial component can extract a local likelihood, and then they are fuzzily polymerized with dynamic weights, so we call the factor as “fuzzy weight factor”, which is m , typically has a value ranging from 0 to 5.

Fig. 4 The whole procedure of face retrieval by dynamically integrating several DNNs. (a) The original images including faces. (b) Facial components detected and located from template and target faces. (c) Feature extraction and classification for each kind of component by corresponding DNN. (d) Classification of different kinds of peculiar components, here the color of column means different kind of components, and the height of column means the weight value of current classification. (e) Vectors of classification results with dynamic weights generated from template and target faces. (f) Calculating the whole similarity for face retrieval.

If the face is similar to the template, the J_m becomes smaller, so the given expression must meet the following conditions:

$$\left\{ \begin{array}{l} J_m(U) = \min(\sum_{i=1}^K U_i^m D_i) \\ \sum_{i=1}^K U_i \leq 1 \\ 0 \leq U_i \leq 1, 1 \leq i \leq K \end{array} \right. \quad (14)$$

Based on Lagrange steepest descent method, the best weight can be calculated as:

$$U_i = \frac{(\frac{1}{D_i})^{\frac{2}{m-1}}}{\sum_{i=1}^K ((\frac{1}{D_i})^{\frac{2}{m-1}})} \quad (15)$$

Based on the preceding operation, the detected face components and template facial components are separately sent into the DNNs before their classification results and weights can be obtained. Thus, the similarity vector is formed as:

$$S_i = [(k_i^M, U_i^M), (k_i^T, U_i^T)] \quad (16)$$

Here $i = 1 \sim 8$ is the index of facial components; k_i^M, U_i^M are the results of detected facial component i and its weight; and k_i^T, U_i^T are the results of template facial component i and its weight. The overall similarity can be calculated as

$$S = \sum_{i=1}^K H(k_i^T, k_i^M) \times U_i^T \times U_i^M \quad (17)$$

H is the judging function of patterns, which is expressed as

$$H = \begin{cases} 1, & \text{if } (k_i^T = k_i^M) \\ 0, & \text{if } (k_i^T \neq k_i^M) \end{cases} \quad (18)$$

Using expression (17), we can obtain the similarity between the detected face and template face. Thus, the retrieval is achieved.

5. Experimental results and analysis

We have tested the new algorithm on a personal computer that has an Intel Core CPU with 3.33 GHz and 8GB DDR.

For full contrast, our experiments are operated on two types of image sets: standard dataset in which the illumination, expression, and angle are strictly controlled; and natural dataset in which the images are captured from wild videos.

5.1 Datasets of standard faces

We test our algorithm on a number of public standard datasets, including

- (1) ORL²⁸, which contains 400 images of 40 subjects taken with varying poses and expressions;
- (2) Extended Yale B database²⁹, which mainly tests illumination robustness of face recognition algorithms, containing 38 subjects, with 64 frontal images per subject taken with strong directional illuminations;
- (3) CMU PIE³⁰, which has the same random partition described in our experiments; and
- (4) Multi-PIE database¹¹, which consists of images of 337 subjects at a number of controlled poses, illuminations, and expressions taken over four sessions. Each standard face image set is partitioned through random selection of half the set per subject for training and the rest for testing. For contrast, several popular algorithms are chosen, such as MKL³¹, learning-based descriptor³², simile classifier³³, background sample³⁴, associate-predict model³⁵, mid-level feature³⁶, visual attributes³⁷, and classic DNN for entire face, which is one of the hot research directions currently. All of these algorithms are used to work on the same set of data. Table 1 presents the results.

Table 1: Retrieval accuracy on standard datasets.

Method	Accuracy(%)
MKL	85.2
Learning-based Descriptor (LD)	92.3
Simile Classifier (SC)	88.7
Background Sample (BS)	90.1
Associate-predict Model (AM)	86.6
Mid-level Feature (MF)	88.3
Visual Attributes (VA)	89.5
DNN for Entire Face (DNN-EF)	91.3
ours (DNNs-Fusion)	91.6

The contrast shows the accuracy of our method is top-ranked on standard datasets.

5.2 Dataset of unconstrained faces

We further test our algorithm on the more challenging faces in the Pubfig image set. Pubfig, built by Columbia University, contains 58,797 images of 200 persons; all of the images are captured from natural videos and pictures³⁷.

Similarly, half of the Pubfig images are used for training and the rest are used for testing. We also choose the same algorithms for contrast. The performances are shown in Fig. 5.

Fig. 5: True positive rate and false positive rate test on the wild image set of Pubfig

In this experiment, our algorithm can achieve higher true positive rate and its false positive rate is lower than that of other algorithms.

5.3 Computational cost

From a computational cost perspective, we see that the overall calculation is related to the number of layers in DNNs because of the convolution operations in the hidden layers. In this paper, our DCDNN consists of eight parallelizable DNNs, each of which has seven layers. Based on the hardware and test on Pubfig mentioned in section 5.2, the mean CPU time used by our algorithm is approximately 0.12 second per image. For comparison purposes, the other algorithms use up the following time: MKL, 0.1 second per image; learning-based descriptor, 0.15 second per image; simile classifier, 0.13 second per image; background sample, 0.12 second per image; associate-predict model, 0.11

second per image; mid-level feature, 0.15 second per image; and visual attributes, 0.14 second per image.

Aggregate analyzing the experiment results, our algorithm has strong advantage that it can spend less time to achieve better treatment effect, especially on unconstrained faces.

6. Conclusion and future work

In this study, we obtain facial components via optimized ASM and then design different DNN models for the different facial components: left brow, right brow, left eye, right eye, nose, mouth, whisker, and visible scars or birthmarks. All of the DNN models are synthesized with different dynamic weights. Such DNN models are trained by the known component sets, after which the unknown face detected from the video or picture, and the template faces, are sent to the DNN models for calculating their similarities. Experiments have demonstrated that through this method, the accuracy of face retrieval can be improved, particularly on natural image sets. Therefore, our method can be applied to actual video surveillance systems as well.

References

1. A. J. O'Toole, P. J. Phillips, S. Weimer, D. A. Roark, J. Ayyad, R. Barwick, and J. Dunlop. Recognizing people from dynamic and static faces and bodies: Dissecting identity with a fusion approach. *Vision Research*, (2011).
2. M. Everingham, J. Sivic, and A. Zisserman. Taking the bite out of automated naming of characters in TV video. *Image and Vision Computing*, 27(5) (2009) 545-559.
3. D. Ramanan, S. Baker, and S. Kakade. Leveraging archival video for building face datasets. In *ICCV*, (2007).
4. B. Xiao, X. Gao, D. Tao, Y. Yuan and J. Li, Photo-sketch synthesis and recognition based on subspace learning, *Neuro Computing*, 73 (2010) 840-852.
5. X. Wang and X. Tang, Face photo-sketch synthesis and recognition, *IEEE Transactions Pattern Analysis Machine Intelligence*, 31(11) (2009) 1955-1967.
6. L. Wolf, T. Hassner, and Y. Taigman. Descriptor based methods in the wild. In *Faces in Real-Life Images Workshop in ECCV*, (2008).
7. D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2) (2004) 91–110.
8. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, (2005).

9. M. Valstar, B. Martinez, X. Binefa, and M. Pantic. Facial point detection using boosted regression and graph models. CVPR, (2010).
10. P. Felzenszwalb, R. Girshick, and D. McAllester. Cascade object detection with deformable part models. CVPR, (2010).
11. R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. Image and Vision Computing, (2010).
12. P. Phillips, H. Moon, S. Rizvi, and P. Rauss. The feret evaluation methodology for face-recognition algorithms. IEEE TPAMI, (2000).
13. Prabhu, U. ; Jingu Heo ; Savvides, M.. Unconstrained Pose-Invariant Face Recognition Using 3D Generic Elastic Models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(10) (2011) 1952-1961.
14. P. Yan and K.W. Bowyer. Biometric recognition using three dimensional ear shape. In IEEE Transactions on Pattern Analysis and Machine Intelligence, 29 (2007) 1297–1308.
15. John D. Bustard and Mark S. Nix. 3D Morphable Model Construction for Robust Ear and Face Recognition. In CVPR, (2010).
16. Li Deng, Three Classes of Deep Learning Architectures and Their Applications: A Tutorial Survey. Microsoft Research, (2013).
17. Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In NIPS, (2007).
18. H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In NIPS, (2007).
19. M. Zeiler, D. Krishnan, G. Taylor, and R. Fergus. Deconvolutional networks. In CVPR, (2010).
20. J. Yang, K. Yu, Y. Gong, and T. S. Huang. Linear spatial pyramid matching using sparse coding for image classification. In CVPR, (2009).
21. Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, Lior Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In CVPR, (2014).
22. Sachin Sudhakar Farfade, Mohammad Saberian, Li-Jia Li. Multi-view Face Detection Using Deep Convolutional Neural Networks. [J]. (2015) 643-650.
23. Florian Schroff, Dmitry Kalenichenko, James Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. In CVPR, (2015).
24. T. F. Cootes, C. J. Taylor, D. H. Cooper, J. Graham: Active Shape Models | their Training and Application. CVIU 61 (1995) 38-59.
25. A. Bandera, J.P.B.J.M. Pe'rez-Lorenzo, and F. Sandoval. Mean Shift Based Clustering of Hough Domain for Fast Line Segment Detection. Pattern Recognition Letters, 27 (2006) 578-586.
26. Dan Cireřan, Ueli Meier, Jonathan Masci, Jürgen Schmidhuber. Multi-column deep neural network for traffic sign classification. Neural Networks: 32 (2012) 333–338.
27. Hugo Larochelle, Yoshua Bengio, Jérôme Louradour, Pascal Lamblin. Exploring Strategies for Training Deep Neural Networks. Journal of Machine Learning Research. (2009) 1-40.
28. ORL Face Set:
<http://www.cl.cam.ac.uk/research/dtg/attarchive/facedata/base.html>
29. A. Georgiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intelligence, 23(6) (2001) 643–660.
30. T. Sim, S. Baker, and M. Bsat. The CMU pose, illumination and expression database. IEEE Trans. on Pattern Analysis and Machine Intelligence, 25(12) (2003) 1615–1618.
31. N. Pinto, J. J. DiCarlo, and D. D. Cox. How far can you get with a modern face recognition test set using only simple features? In CVPR, (2009).
32. Z. Cao, Q. Yin, X. Tang, and J. Sun. Face recognition with learning-based descriptor. In CVPR, (2010).
33. Kumar, N., Berg, A.C., Belhumeur, P.N., Nayar, S.K.: Attribute and Simile Classifiers for Face Verification. ICCV, (2009).
34. Wolf, L., Hassner, T., Taigman, Y.: Similarity Scores based on Background Samples. ACCV, (2009).
35. Q. Yin, X. Tang, and J. Sun. An associate-predict model for face recognition. In CVPR, (2011).
36. Y.-L. Boureau, F. R. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In CVPR, (2010).
37. Neeraj Kumar, C. Berg, Peter N. Belhumeur, and Shree K. Nayar. Describable Visual Attributes for Face Verification and Image Search. IEEE Transactions On Pattern Analysis And Machine Intelligence. 33(10) (2011) 1962-1977.