Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 9, September 2014, pg.314 – 323

RESEARCH ARTICLE

Face Image Retrieval Using Pose Specific Set Sparse Feature Representation

Abdul Afeef N[1], Sebastian George[2]

[1]Department of Computer Science, Viswajyothi College of Engineering and Technology Kerala, India

[2]Assistant Professor of Computer Science, Viswajyothi College of Engineering and Technology Kerala, India

afeefengg@gmail.com

Abstract— There are largely available consumer photos are available in our life, among those a big percentage of photos are photos with human faces. Thus content based face image retrieval is an emerging technology for many applications. A new method for content based face image retrieval is proposed. Given a query face image, content based face image retrieval find similar faces from the database. There were mainly three steps- preprocessing, feature extraction and face retrieval. Preprocessing included face detection and landmark detection. After preprocessing, extract pose specific uniform LBP features from the face. Novel set sparse feature representation is used to represent extracted pose specific LBP feature as sparse code using a pre-defined set. After 15 attributes from face is used to make face representation more distinguish. Finally set sparse feature representation is used as inverted index in face retrieval to find similarity score of query face image with database face images. It has applications in automatic face annotation, crime investigation etc. Top results related to a query face image with existing method and memory usage were analyzed. Experimental results shows that proposed method have better top results and efficient in memory usage compared to existing method.

Keywords— Face image retrieval, sparse feature representation, pose specific feature

I. INTRODUCTION

Due to the popularity of digital devices and the rise of social network and photo sharing services (e.g., Facebook, Flickr), there are largely growing consumer photos available in our life. Among all those photos, a big percentage of them are photos with human faces. The importance and the sheer amount of human face photos make manipulations (e.g., search and mining) of large-scale human face images a really important research problem and enable many real world applications. The main goal is to address one of the important and challenging problems, content-based face image retrieval. Given a query face image, content-based face image retrieval tries to find similar face images from a large image database. It is an enabling technology for m any applications including automatic face annotation, crime investigation.

Several techniques are proposed in recent years for face image retrieval which uses different face feature representation and face retrieval strategy. Face matching and retrieval using soft biometrics [2] uses soft biometrics embedded in face such as gender and facial marks and this information combined with face matching score. Scalable face image retrieval with identity based quantization and multi-reference re-ranking [3] uses new scalable face representation using both local and global features. It uses a new multi-reference distance to re-rank candidate images using the hamming signature. Semi-supervised face image retrieval using sparse coding with

identity constraints [4] integrate partial identity information in sparse feature representation for better face image retrieval results. Combining attributes and fisher vectors [5] improves the performance of retrieval of particular objects as well as categories. Scalable face image retrieval using attribute enhanced sparse codewords uses both low level features such as uniform LBP features and high level human attributes for better retrieval results.

Traditional methods for face image retrieval usually use low-level features to represent faces, but low-level features are lack of semantic meanings and face images usually have high intra-class variance (e.g., expression, posing), so the retrieval results are unsatisfactory. To tackle this problem, use identity based quantization and identity-constrained sparse coding, but these methods might require clean training data and massive human annotations. Many of the face image retrieval techniques does not consider pose of the face for extracting features from the face. So same person of different poses may represented by dissimilar features and different classes. Traditional sparse coding for face image retrieval represent a feature by a linear combination of column vectors of a dictionary. Number of sparse codewords depends on size of the dictionary and sometimes it may resulted in representing same person in different classes.

The main objective of this work is to provide a new perspective on content-based face image retrieval by incorporating pose specific extraction of components and features from face. Pose specific uniform LBP feature is extracted from the face which is then represented as sparse codewords. A novel representation called set sparse feature representation is proposed to represent extracted pose specific uniform LBP features from the face using a pre-defined set. 15 attributes are used to make represented set sparse feature more distinguish. In retrieval stage, set sparse code is inversely indexed with image list to find similarity score for better face image retrieval. The main characteristics of proposed methods are 1) Pose of the face is considered in feature extraction and 2) Set sparse feature representation is used to find sparse code of extracted features.

II. OVERALL DESIGN OF PROPOSED METHOD

The overall design of proposed face image retrieval technique is shown in Figure 1. Initilally select the query image and preprocessing such as face detection, 66 landmark detection and face alignment are done for the query face image. After preprocessing, find pose of the face image such as left or right. Then extract pose specific four components from the face such as eyebrow, eye, nose and mouth. In feature extraction step, compute 59 dimensional uniform LBP feature from components and select 15 attributes of face image. Then novel set sparse feature algorithm is used to create representative sparse code for the face image and attributes are used for better distinguishing power. Finally in retrieval stage, sparse code inversely indexed with image list and compute similarity score with database images after finding hamming distance of query with database images.

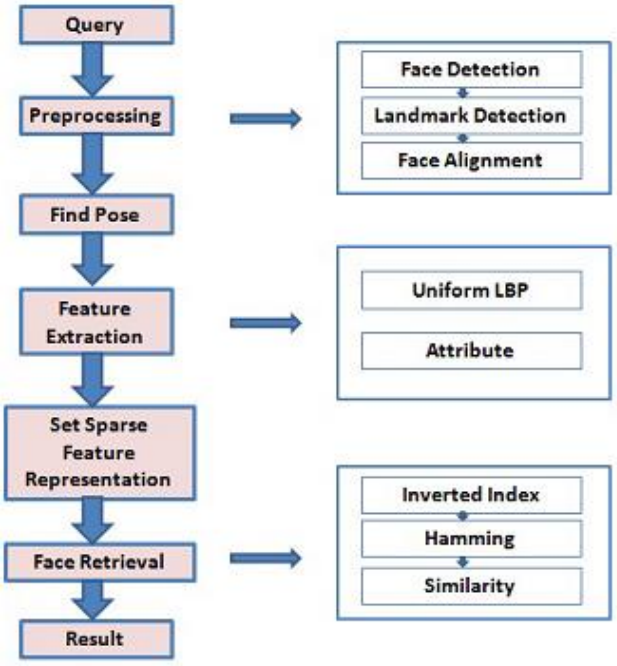

Fig 1 Overall design of proposed method

III. MODULE DESCRIPTION

The system includes four main components

1. Select Query Image.
2. Preprocessing.
3. Find Pose.
4. Feature Extraction.
5. Set Sparse Feature Representation.
6. Face Retrieval.

1. Select Query Image

Content based face image retrieval aims to retrieve similar face images from the database of given query image. Initially select a query image which needs to retrieve similar face images from database. Figure 2 shows the selected query image.

Fig 2 Query image

2. Preprocessing

After selecting query face image, preprocessing is done to query image. It mainly consist of three steps such as face detection, landmark detection and face alignment.

Use Viola-Jones face detector to locate face in the query image. It returns a four dimensional row vector which gives the information about top left corner of the face, width and height of face from top left corner. First value represent x coordinate of the top left pixel of face and second value represent y coordinate of that pixel. Third and fourth value represent width and height respectively from top left corner of the face. Using this, find four corners of the face. Using these four corners, a rectangular box is drawn around face. Figure 3 shows corresponding face for the query image in Figure 2.

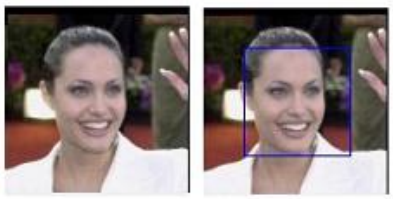

Fig 3 Detected face

After face detection, 66 landmarks are located in detected face image. It uses active shape model to locate landmark. It returns 132 dimensional column vector which contain x and y coordinates of detected landmarks. These landmarks are used for finding pose and extracting pose specific components such as eyebrow, eye, nose and mouth. Figure 4 shows detected landmarks of detected face image in Figure 2.

Fig 4 Landmark detection

After locating 66 landmarks from detected face image, face is aligned using the detected landmarks. For that, boundary landmarks are selected and store the x and y coordinates of boundary points in different column vector. Then extract largest possible polygon that can extract using boundary points as aligned face image. Figure 5 shows corresponding aligned face image for the landmark detected face in Figure 2.

Fig 5 Aligned face

3. Find Pose

The proposed method considers pose of the query face image to extract components from the face. It uses four landmarks from the 66 located points. Let the difference between x coordinates of landmark 48 and 4 be 'A' and difference between x coordinates of 12 and 54 b e 'B' . If the value of A greater than B, then set the pose as right, else set the pose of the query face image as left. This pose is then used in the feature extraction stage to extract pose specific component extraction from the face such as eyebrow, eye, nose and mouth. Figure 6 shows the pose of query face image.

Fig 6 Right pose

4. Feature Extraction

After finding pose of the query face image, extract pose specific components from the face. If pose equals right, then extract right eyebrow, eye, nose and mouth. Figure 7 shows extracted pose specific components from query face image. Then extract 59 dimensional uniform LBP using algorithm1. Then select 15 attributes [7] of query face image for set sparse feature representation and finally for the indexing stage. Selected attributes are, male, white, youth, senior, black hair, blond hair, bald, no eyewear, mustache, chubby, straight hair, receding hairline, bushy eyebrows, big nose and big lips.

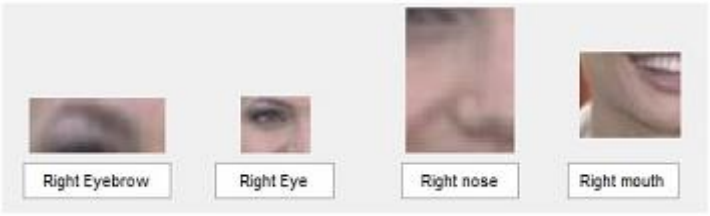

Fig 7 Pose specific components

Uniform LBP is defined for a non-boundary pixel in a patch by using eight neighbouring pixels around it. Let 'c' be the centre pixel and p_1 to p_8 be eight neighbouring pixels from left bottom in clockwise direction. Compare centre pixel 'c' with each p_i, i from 1 to 8. If $p_i > c$, set $a_i = 1$, else 0. Then orderly arrange all a_i's and find number of transition in bits. If transition less than three, it considered as uniform and its uniform LBP value is set as decimal value of a_i from 1 to 8. If transition greater than two, it considered as non-uniform and set uniform LBP value of centre pixel as a constant say for example 5. In a 10x10 patch, there are 64 non boundary pixels and found uniform LBP value of all 64 non boundary pixels and compute the histogram and then get 59 dimensional uniform LBP value for a patch. Likewise find 59 dimensional ULBP from all square patches of size 10x10 from all extracted pose specific components. Algorithm1 is used to compute 59 dimensional ULBP from all extracted pose specific components which uses the inputs extracted components and the pose of the face gives the output ULBP59 as 59 dimensional ULBP for the face.

Algorithm1 : Compute-59dimULBP

Input : Extracted Components, pose

Output : ULBP59

1. Start
2. InitializeprevULBP59as empty
3. Select components
    (a) If pose equals right, select right components and convert to gray
    (b) Else select left components and flip using algorithm2
4. For each selected components (order : eyebrow, eye, nose, mouth)
    (a) Extract all 10x10 square patches row-wise
    (b) For each square patch
        i. Compute ULBP values for all 64 non-boundary pixels
        ii. Compute histogram of ULBP
        iii. Append histogram with prevULBP59
5. SetULBP59equalsprevULBP59
6. End

If the pose of face is left, then it needs to flip the extracted left components before extracting ULBP. Algorithm2 flip the extracted let components as look like right components for the standardisation of component shape for all database images.

Algorithm2 : Flip

Input : Extracted Left Components

Output : Flipped Left Components

1. Start
2. For each input (order : left eye brow, eye , nose and mouth)
    (a) Convert to gray image
    (b) Find number of rows and columns
    (c) Set *n* as number of columns and *m* as rows
    (d) Initialize *Flipped* as zero matrix of *m* rows and *n* columns
    (e) For each column *i* from *1* to *n*
        i. Set column *i* of *Flipped* as column(*n-i+1*) of input
3. Set Flipped Left Components as set of outputs from step2
4. End

5. Set Sparse Feature Representation

Sparse coding is used for representing feature as zeros and ones which contains most of the elements as zero. It is very useful in retrieval stage in which inverted indexing is performed on sparse code versus image list. Traditional sparse coding method required a dictionary which used to represent a feature by linear combination of column vectors of dictionary. This work proposed a new perspective on sparse coding which uses a five dimensional set for representing a feature as sparse called set sparse feature representation. It does not need a dictionary. By combining with attributes from face, it retrieves better top results compared to base paper and it also very effective in memory. Algorithm3 explains the novel set sparse feature algorithm which takes input

ULBP59 from algorithm1 and gives back the result *set sparse feature* as output. *set sparse feature* contains most of the elements as zero and others as ones.

Algorithm3 : Set Sparse Feature Face

Input : ULBP59

Output : Set Sparse Feature

1. Start
2. Find number of rows *m* and columns *n* of *ULBP59*
3. Combine adjacent two columns of *ULBP59*(So *n/2* columns)
4. Assign set as [1 2 10 15 20]
5. Initialize *Set Sparse Feature* as empty
6. For *i* increment from 1 to n/2
    - 6.1 For *j* increment from 1 to m
        - 6.1.1 Set *num* as*ULBP59(i,j)*
        - 6.1.2 Initialize *temp* as empty
        - 6.1.3 For *i* decrement from 5 to 1
            - 6.1.3.1 Set *Curr* = Set(i)
            - 61.3.2 Check *flag* as *num* >= *Curr*
                - 1) If`*flag* equals 1
                    - i. Append 1 to starting of *temp*
                    - ii. Set *num* as reminder of *num*/*Curr*
                    - iii. Set *Curr=Set(i-1)*
                - 2) Else
                    - i. Append 0 to starting of *temp*
                    - ii. Set *Curr = Set(i-1)*
                    - iii. Goto step 6.1.3
        - 6.1.4 Append *temp* to *Set Sparse Feature*
        - 6.1.5 If *j* < *m*, goto step 6.1, else goto step 6
7. End

Algorithm4 is used for further increasing the distinguishing power of the extracted *set sparse feature* by using the 15 attributes from face. It takes the inputs *set sparse feature* and 15attributes and gives back final *Attr_Set_Sp_Code* which is used in the final retrieval stage for inverted indexing.

Algorithm4 : Attribute Set Sparse Face

Input : set sparse feature , 15 Attributes

Output : Attr_Set_Sp_Co de

1. Start
2. For *i* increment from 1 to 15
    - (a) If positive attribute f_i equals 1
    - (b) Else f_i equals 0

3. Select *n* attributes say a_1 to a_n from f_1 to f_{15}(1<= *n* <= 15)
4. Set the length of`*set sparse feature* equals *m*
5. Append rem(*m*/*n*) zeros at the end of *set sparse feature*
6. Set new length of *set sparse feature* equals *m_new*
7. Divide *m_new* into equal *n* partitions of length *len*
8. Set *zero_append* as zero vector of size *len*
9. Initialize *temp* as empty matrix
10. For each partition *i* increment form 1 to *n*
    (a) Set *Add_part* as partition *i*
    (b) If a_i = 1, append *zero_append* at the end of *Add_part*
    (c) Else append *zero_append* at the start of *Add_part*
    (d) Append *Add_part* at the end of *temp*
11. Set *attr* as row vector f_1 to f_{15}
12. Set *attr_inv* as 1- *attr*
13. Set *attr_temp* as empty
14. For *i* increment from 1 to 80
    (a) If`mod(i,2) equals 1, append *attr* at end of`*attr_temp*
    (b) Else append *attr_inv* at end of`*attr_temp*
15. Append *attr_temp* at the end of *temp*
16. Set final *Attr_Set_Sp_Code* as *temp*
17. End

6. Face Retrieval

It mainly consist of three steps. Inverted indexing, hamming distance calculation and similarity score calculation. In inverted indexing, sparse code is inversely indexed with image list. For the computed set sparse code for the query face image, we check the matched sparse codes of the input query. Then we list all the images from the inverted index which matches the sparse codes of the query face image. Then hamming distance is calculated for images from the inverted index by XORing the input 15 binary attributes with 15 binary attributes from the images in inverted index. Then finally similarity score is calculated for the images which have hamming distance less than a threshold, threshold can change from 0 to 15. For the effective retrieval threshold is set to 4.

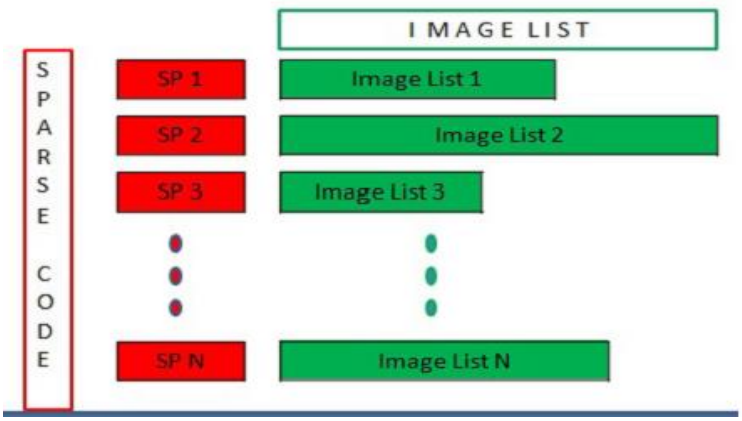

Fig 8 Inverted index

Figure 8 shows the inveted indexing of sparse code with image list. Hamming distance is calculated by counting number of ones in the XOR ed output of input with database sparse code matched images from inverted index. Figure 9 shows the computation of hamming distance.

$$S(i,j) = \begin{cases} Number\ of\ ones\ in\ c_i \cap c_j, & if\ hamming \leq threshold \\ 0, & otherwise \end{cases} \quad (1)$$

Input attribute : 11111 00000 11111
DB images : 11110 00100 11110
XOR : 00001 00100 00001
Hamming Dist = No: of 1's in XOR = 3

Fig 9 Computation of hamming distance

Equation 1 shows computation of similarity score. S(i,j) is the similarity score between image i and j. c_i and c_j are the sparse code for the face image i and j. Finally retrieve top 10 similarity scored images as the result of query face image. Figure 10 shows the retrieval results corresponding to query face in Figure 2.

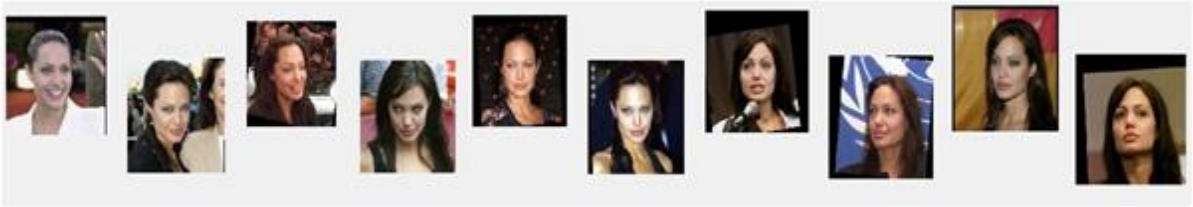

Fig 10 Retrieval result

IV. EXPERIMENTAL RESULTS

The proposed content based face retrieval technique has been evaluated and compared with an existing content based face retrieval method [1] in the same scenario. The objective is to evaluate the accuracy of retrieving top results and memory utilization for storing sparse code using the proposed technique with respect to existing method. All the algorithms were implemented using MATLAB.

Error rate analyses the number of false retrieval related to query image. For this, consider 500 results obtained from 50 query images in the database. 50 images consist of 5 persons and each person has 10 images each. Each of 50 images given as query image and analysed retrieval results. It also analysed top n results, where n varies from 1 to 10. Error rate is calculated using the equation 2. Then related improvement in top results is analysed using the equation 3. Figure 11 shows error in to results comparison between existing [1] and proposed method.

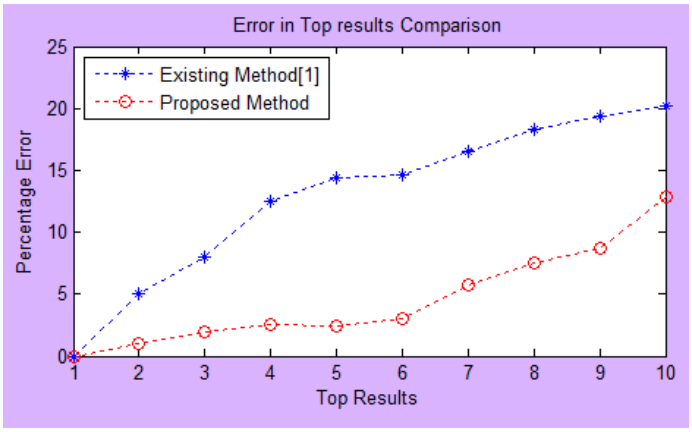

Fig 11 Error in top results comparison

$$Error\ TOP(N) = \frac{Number\ of\ false\ detection\ in\ 50\ image's\ retrieval}{50*N} * 100 \qquad (2)$$

$$Rel\ Impr\ TOP(N) = \frac{Error\ TOP(N)[1] - Error\ TOP(N)prop}{Error\ TOP(N)[1]} * 100 \qquad (3)$$

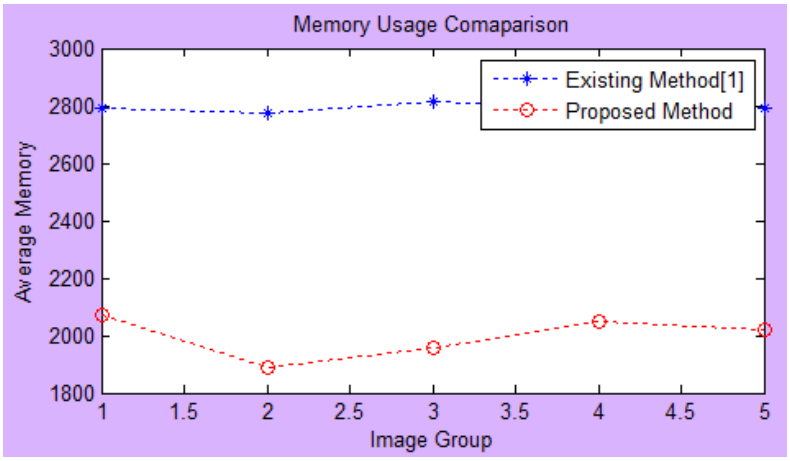

Fig 12 Memory usage comparison

Analysing memory aims to count number of sparse codes for storing the face. We analyse average number of sparse codes needed for storing each person's face in the database and compared with existing method. Experimental results shows that proposed method have better memory utilisation compared to existing method [1]. Average number of sparse code for a person is calculated using equation 4 where *Sparse(i)* is the sparse code for ith face image. Figure 12 shows memory usage comparison between existing [1] and proposed method.

$$Sparse\ Average(N) = \frac{\sum_{i=(N-1)*10+1}^{(N-1)*10+10} Number\ of\ ones\ in\ Sparse(i)}{10} \qquad (4)$$

V. CONCLUSIONS

An efficient content based face image retrieval system is proposed which uses pose specific component extraction for feature computation. It also proposed a novel set sparse feature algorithm to represent extracted feature as sparse code using a set. Attributes from face are used to further improving the distinguishing power in sparse representation. Finally inverted index is used in retrieval stage. It has applications in automatic f ace annotation, crime investigation etc. This work analysed top results related to a query image with existing method and memory usage. Experimental results shows that proposed method have better top results and efficient in memory usage compared to existing method.

REFERENCES

[1] Bor-Chun Chen, Yan-Ying Chen, Yi n-Hsi Kuo, and Wi nston H. Hsu, "Scalable Face Image Retrieval Using Attribute-Enhanced Sparse Codewords", in IEEE Trans on Multimedia, vol. 15, no. 5, Aug 2013.

[2] U. Park and A. K. Jain, "Face matching and retrieval using soft biometrics", in IEEE Trans. Inf. Forensics Security, vol. 5, no. 3, pp. 406-415, Sep 2010.

[3] Z. Wu, Q. Ke, J. Sun, and H.-Y. Shum, "Scalable face image retrieval with identity-based quantization and multi-reference re-ranking", inProc. IEEE Conf. Computer Vision and Pattern Recognit, 2010.

[4] B.-C. Che n, Y.-H . Kuo,Y.-Y. Chen, K.-Y. Chu, and W. Hsu, "Semi-supervised face image retrieval using sparse coding with identity constraint", in in P roc. ACM Mul -timedia, 2011.

[5] M. Douze, A. Ramisa, and C. Schmid, "Combining attributes and fisher vectors for efficient image retrieval", inProc. IEEE Conf. Computer Vision and Pattern Recognit. 2011.

[6] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification", in Proc. Int. Conf. Computer Vision, 2009.

[7] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Describable visual attributes for face verification and image search", in IEEE Trans. Pattern Anal. Mach. Intel l., Special Issue on Real-World Face Recognition vol. 33, no. 10, pp. 1962-1977, Oct. 2011.

[8] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in Proc. IEEE Conf. Computer Vision and Pattern Recognit., 2001.

[9] S. Milb orrow and F. Nicollsl, "Locating facial features with an extended active shape model", inProc. Eur. Conf. Computer Vision, 2008.

[10] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification" in Proc. IEEE Conf. Computer Vision and Pattern Recognit.,2009.

[11] A. Coates and A. Y. Ng, "The importance of encoding versus training with sparse coding and vector quantization", inProc. ICML, 2011.

