Neural Comput & Applic (2012) 21:683–694
DOI 10.1007/s00521-011-0665-8

ICONIP2010

Machine learning approach for face image retrieval

Dianhui H. Wang · Paul Conilione

Received: 16 April 2011 / Accepted: 7 June 2011 / Published online: 22 June 2011

Abstract Face image retrieval (FIR) is useful to many domain applications, such as helping police to catch criminals or managing householders. However, little research has been done that uses individual face features for image comparison and retrieval. This paper aims to develop a machine learning approach for face image retrieval based on the local face features of the eyes, nose, and mouth. Neural networks are used to localise facial features, and to implement a learning pseudo metric (LPM) to filter out irrelevant images for retrieval efficiently based on semantic information. Our FIR system performs below average given traditional performance measures, but inspecting actual retrieved images it shows strong promise. It is observed that the LPM semantic filtering method was found to reduce the database size by up to 50% without a significant reduction in retrieval performance.

Keywords Face image retrieval · Semantic filtering · Learning pseudo metrics · Neural networks

1 Introduction

Content-based image retrieval (CBIR) aims to retrieve images from a database that match a users' query based on image content. The past decade has seen rapid development in CBIR sophistication and performance [1], with many techniques being developed to solve problems in medical imaging, art, and web-based image retrieval. In the special case of face image retrieval (FIR), a user wishes to find some similar faces in a database for a given query. The user can submit a query in the form of an image (query-by-example) [2], sketch [3], or description [4]. FIR has a wide range of potential applications, such as being used to search personal photograph albums for friends and family [5], used in law enforcement agencies needing to search large databases of faces for suspects [6] or can be integrated into Internet search engines [7].

CBIR systems map images to the feature space using low-level features such as color, texture, and shape. The distance between any two points in the feature space is measured using a metric function, such as Euclidean, Mahalanobis, or City Block. The distance is used to rank the database images against the query image. However, it has been found that metrics, such as Euclidean distance, may not be able to reflect the similarity in terms of semantics. For example, two images can be far apart in the metric space using the Euclidean metric but are semantically very close [8]. There is usually no direct relationship between the low-level features of an image with the semantic concepts that a person would associate with the image. This problem is known as the "semantic gap" [9]. A wide range of semantic-based methods have been developed to overcome the semantic gap problem and improve image retrieval including machine learning, relevance feedback, semantic templates, and object ontologies [10]. It has been shown that the machine learning-based learning pseudo metric (LPM) has the ability to more accurately determine the semantic similarity between multimedia objects compared with conventional metrics [11]. The LPM determines whether any two instances are semantically related. However, it does not provide a measure of how similar the two instances are, only that they

D. H. Wang (✉) · P. Conilione
Department of Computer Science and Computer Engineering, La Trobe University, Melbourne, VIC 3086, Australia
e-mail: dh.wang@latrobe.edu.au
URL: http://homepage.cs.latrobe.edu.au/dwang/

belong to the same semantic class or not. Hence, we propose to use the LPM as a semantic filter for removing face images from a database that are not strongly related semantically to the query image.

The common approach for FIR systems is to extract a feature vector that describes the entire face appearance. The classic unsupervised subspace learning approach called Eigenfaces, applies Principle Component Analysis (PCA) to well-framed images [12]. While the supervised subspace learning method called Fisherface uses the Fisher's Linear Discriminate (FLD), which considers the classes that each training example belongs to and can lead to better separation of the faces in the fisherspace [13]. More recent developments include the Orthogonal-Laplacianface, which uses the Orthogonal Locality Preserving Projection (OLPP) method to provide greater discriminating power than the standard Locality Preserving Projection [14]. While the semi-supervised dimensionality reduction using relevance feedback from the user, called Maximum Margin Projection, has provided strong retrieval results [15]. The problem with using global feature extraction methods is that retrieval performance can be affected by changes in face appearance, e.g. illumination, pose, expression [16]. It has been shown that identifying faces by their face features (eyes, nose) provides computational and accuracy benefits [17].

This paper aims to develop a local face feature-based FIR system using neural networks, where a LPM-based semantic filter is performed before using the Euclidean distance measure for final image comparison and ranking. The key modules in our FIR system include learning localization of the local face features, face data representation using local features, semantic filtering, and refined ranking using Euclidrean distance measure within the reduced database.

The remainder of the paper is structured as follows. Section 2 presents a detailed description of our semantic filter FIR system. Section 3 details the experiment parameters and implementation issues for evaluating the FIR performance. Section 4 presents the results and discussion of the semantic filtering-based FIR system. Section 5 concludes the paper with suggested future research directions.

2 System description

2.1 System architecture

FIR system consists of a two-stage retrieval process. Initially, face features are automatically detected using a neural network-based localization system [18]. Followed by feature extraction of the detected face feature image yielding four features per face image $X_i = (X_i^{\text{left eye}}, X_i^{\text{right eye}}, X_i^{\text{nose}}, X_i^{\text{mouth}})$. The same localization and feature extraction steps are applied to the query image yielding $X_q = (X_q^{\text{left eye}}, X_q^{\text{right eye}}, X_q^{\text{nose}}, X_q^{\text{mouth}})$. The first stage of retrieval utilizes a LPM to compare the query image to each database image using the concatenated face feature vectors, (X_i, X_q). The LPM model performs as a semantic classifier and gives a real-valued output, λ, indicating the smilarity between the input images in terms of semantics. If the LPM output value is below a threshold, then the database image is kept for the second stage of retrieval. The aim of the LPM classifier is to remove database images that are semantically dissimilar to the query image. After the database has been filtered, the Euclidean distance measure is used to determine the similarity between the query image and the remaining images in the filtered database as shown in Fig. 1.

2.1.1 Data preprocessing

The face images used for our experiments need to be prepared for three different components of the system. The first dataset is a set of manually extracted face feature images for the off-line training of the neural network for face feature localization. The second dataset is used to train

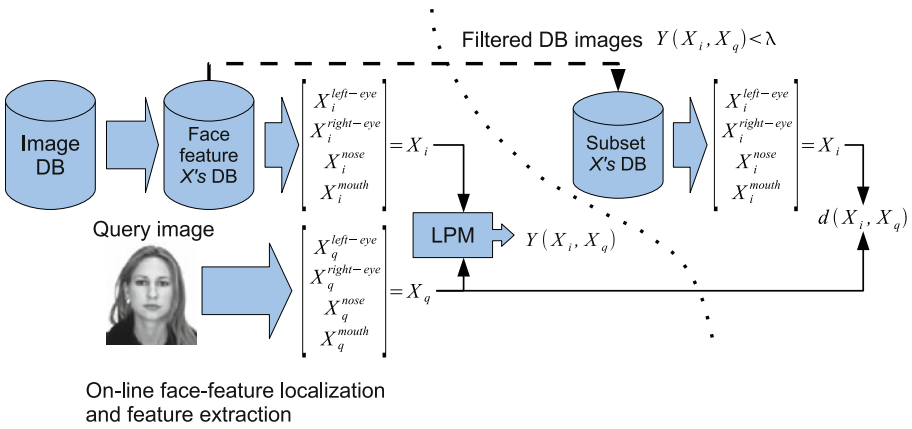


Fig. 1 An overview of the LPM semantic filtering of the database of image. Followed by the comparison of the query to the subset of database instances using a distance measure

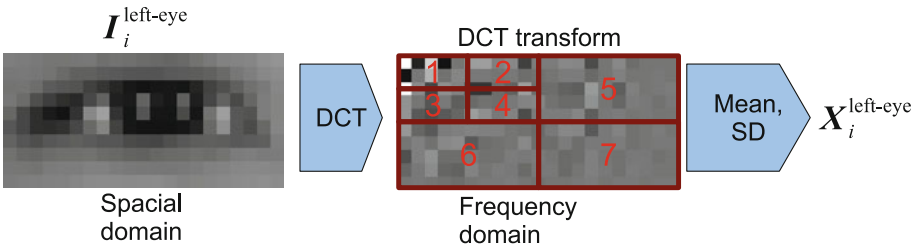


Fig. 2 An image of a face feature (left eye) has 2D DCT applied and the mean and standard deviation (SD) is taken from the seven segments to create $\mathbf{X}_i^{\text{left eye}}$

the LPM classifier and is derived from the automatically extracted face features of the localization stage. The third dataset is used for FIR evaluation and is a collection of face images with associated feature vectors of each face feature.

2.1.2 Face feature localization

Our face feature localization system uses a neural network trained to detect a specific face feature location by scanning a predefined area of a face image. If the neural network output is sufficiently high enough for a given region, the location is registered as the detected face feature and the face feature image is extracted. Feature extraction is then applied to the localized face feature image (small blocks).

2.1.3 Semantic filtering

To perform semantic filtering, we utilize the LPM as a classifier to determine whether any two images belong to the same semantic class or not. The LPM compares two images using the concatenated feature vectors from the corresponding face features. If the LPM output for a database image is below a threshold, λ, then the database image is used for the second stage of retrieval, otherwise it is deemed not to be semantically related to the query image.

2.1.4 Face image retrieval

The Euclidean distance measure is then employed to compare the query image and the images from the filtered database obtained by the LPM semantic filter.[1] The resulting images are then ranked according to ascending similarity scores.

2.2 Image feature extraction

For image feature extraction, we applied the Discrete Cosine Transform (DCT) to the face feature images to yield a frequency domain image. To reduce the dimensionality, the mean and standard deviation of the DCT coefficients in the seven segments as illustrated in Fig. 2 were calculated [19]. These values were combined into a 14-element feature vectors. To train the learner models, we normalized these feature values to the range [−1, 1] using (1) below.

$$X'_{i,j} = \frac{X_{i,j} - X_j^{\min}}{X_j^{\max} - X_j^{\min}}, \tag{1}$$

where $X_{i,j}$ is feature j of image i, $X_j^{\min}$ and $X_j^{\max}$ are the minimum and maximum values, respectively, from all scanned images, and $X'_{i,j}$ is the normalized feature value.

2.3 Face feature localization

To locate face features (eyes, nose, mouth) in a 2D face image, we trained neural networks for detecting the face feature location. Each face feature is treated independently, such that each classifier is trained to only identify a specific face feature. Figure 3 illustrates the overall process of locating a face feature in an image, in this case the left eye using a trained neural network as the pattern classifier.

An input image is scanned with a sliding window within a predefined area to reduce processing time. Feature extraction is applied to each window, and the resulting feature vector is normalized to the range [−1, 1] before being passed to the trained classifier. A classification map is produced after processing the input image, and the location of the face feature is selected by locating the

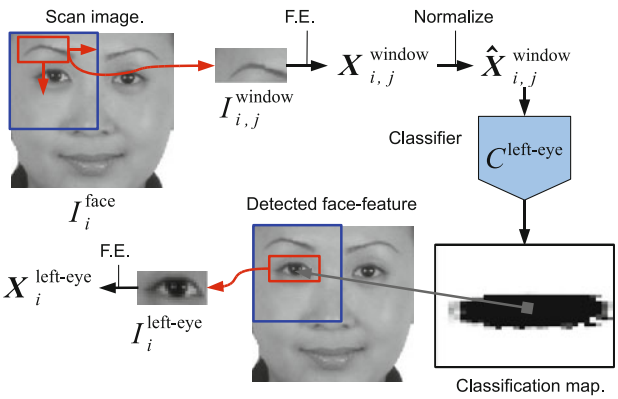


Fig. 3 Face feature localization and feature extraction system overview, note the classification map has had the z-axis reversed to allow for easier reading

[1] Any distance measure can be used in this stage.

sliding window that produced the minimum classifier output. Feature extraction is applied to the window of the detected region to produce a feature vector, which is used for later image retrieval. The following sections detail the system design.

Face feature localization process To detect the location of the face feature, the face image $\mathbf{I}_i^{\text{face}}$ is scanned by a sliding window (the window size is the same size as $\mathbf{I}^f$ which was used for creating the training set). Feature extraction is applied to each image window $\mathbf{I}_{i,j}^{\text{window}}$ where j is the window index, to produce $\mathbf{X}^{\text{window}}$, which is passed to the classifier, and the output $y_{i,j}^{\text{window}}$ is recorded to produce a classification map $\mathbf{Y}_i^f$. The classifier output values closer to 0 are more likely to contain the face feature in the image, and values closer to 1 are unlikely to contain the face feature. It is important to note that the classifier output is not a label of classification. Rather, because the classifier output is in the range [0, 1] (as our target values are $T^{f^+} = 0, T^{f^-} = 1$), we interpret the output as a degree of classification. To predict the position of the face feature, we find the minimum value in the classification map $\mathbf{Y}_i^f$ and locate the corresponding sliding window, which produced the y value in the face image $\mathbf{I}_i^f$.

For an upright, fall frontal, well-framed face image, the facial features appear in predictable locations. When scanning a face feature image for a particular face feature, the whole image is not scanned, rather a limited area is selected to reduce the processing time required. Figure 4 shows the regions selected for each face feature used in our experiments. The sub-regions were sized as a percentage of image width (w) and height (h), where the sub-region for the eyes were $0.5w \times 0.5h$, nose $0.6w \times 0.7h$, and mouth $1.0w \times 0.5h$. The sliding window is moved in a raster scan fashion, where it is incremented by one pixel as it move across the image and once it reaches the end it is moved down by one pixel to repeat the same process.

2.3.1 Generating training samples

To train the face feature localizers, we construct a set of positive (face feature, f^+) and negative (non-face feature, f^-) examples with associated target values. We begin with a set of M face images $\mathbf{I}^{\text{face}} = \{\mathbf{I}_1^{\text{face}}, \ldots, \mathbf{I}_M^{\text{face}}\}$. Let $\mathbf{I}^{f^+} = \left\{\mathbf{I}_1^{f^+}, \ldots, \mathbf{I}_N^{f^+}\right\}$, and $\mathbf{I}^{f^-} = \left\{\mathbf{I}_1^{f^-}, \ldots, \mathbf{I}_N^{f^-}\right\}$ be the sets of face feature images and non-face feature images, respectively. They are extracted for a given face feature $f^+ \in \{\text{left eye, right eye, nose, mouth}\}$ and $f^- \in \{\text{non-left eye, non-right eye, non-nose, non-mouth}\}$. Feature extraction (FE) is applied to the segmented image examples and is combined with a target value to create the classifier training dataset.

To extract the face feature images $\mathbf{I}^{f^+}$, a window is positioned so that it envelops the face feature, then feature extraction is applied to the face feature image to generate the feature vector $\mathbf{X}^{f^+}$. For each non-face feature, we randomly selected 10 negative examples from each face image. Figure 5 illustrates the face feature image and the non-face feature images being extracted from a single face image.

As there are 10 negative examples for each positive example the training dataset is unbalanced. It is well known that unbalanced classes in the training dataset can result in poor learner performance, especially when the data is noisy [20]. Hence, we created additional positive examples by adding a small amount of uniform noise to the feature vectors of the positive examples so that we had equal numbers of positive and negative examples. Such that $\mathbf{X}' = \mathbf{X} + \sigma\mathbf{M}_{\text{Noise}} . * \mathbf{X}$, where $\mathbf{M}_{\text{Noise}}$ is a matrix of the same dimensions as the feature vector, $\mathbf{X}$, and contains uniformly distributed random values in the range $[-1, 1]$ and σ is a percentage coefficient which was assigned randomly a value in the range [0.01, 0.05] for each feature vector that is perturbed. Finally, each feature vector is assigned a target value of $T^{f^+} = 0$ for face features and $T^{f^-} = 1$ for non-face features. The dataset creation process is shown in Fig. 6.

To enhance the generalization capability of the classifier, we perturbed the cropped images by mirroring the images, then rotating the original and mirrored images by $\pm 5°$ so increasing the number of face images in the dataset by a factor of 6.

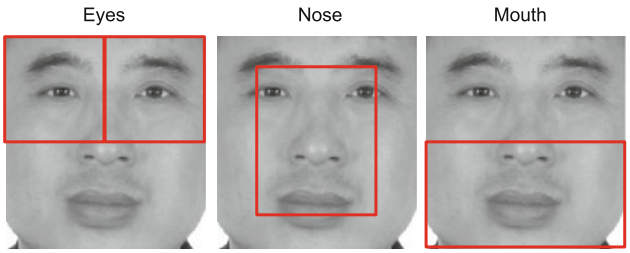


Fig. 4 Each face feature was scanned for within a sub-region of the image to reduce processing time

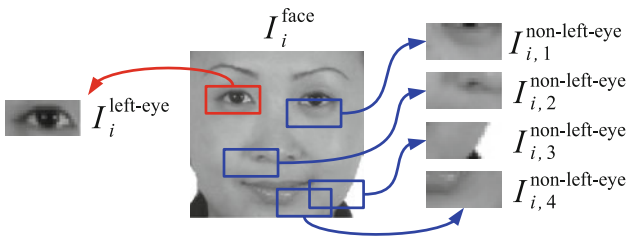


Fig. 5 Example of the left eye image being extracted along with four random images for non-left eye examples

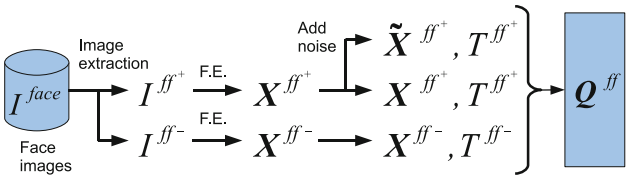


Fig. 6 Process for creating the training set for the face feature localization classifier

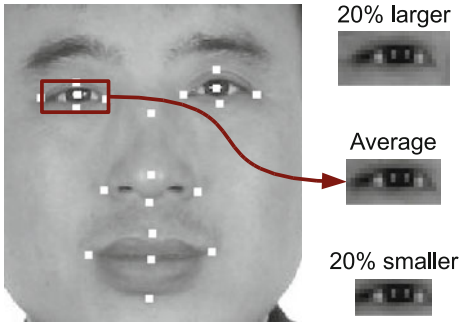


Fig. 7 Example of the three different window sizes for the eye

We also tried face feature window sizes that were larger and smaller than the average face feature window sizes to observe the effect on localization performance. The average window size for each face feature was scaled by ±20%. Figure 7 shows examples of the three different face feature window sizes ($s = \{\text{small, ave, large}\}$) used in our experiments.

2.3.2 Training learner models

Any classifier has a set of parameters, ζ, that determine it's architecture, training algorithm, and ultimately learning performance. To determine the optimal parameters as well as to observe the effect of the face feature window size s, on locating face features, the classifiers were trained with a set of parameters ζ_j using the hold-out method and the classification accuracy was noted. From the dataset $\mathbf{Q}^{f,\,s}$ we produce $\mathbf{Q}^{f,\,s,\,\text{train}}$ and $\mathbf{Q}^{f,\,s,\,\text{test}}$ with a 80%/20% random split, respectively.

Each trained classifier was then used to scan $\mathbf{AR}^{\text{face, Train}}$ and $\mathbf{AR}^{\text{face, Test}}$ images, and the face feature localization accuracy ρ (defined in Sect. 2.3.3) is calculated. The set of parameters ζ^* and the face feature window size s^* that produced the highest localization accuracy were selected for final evaluation using tenfold cross-validation. Such that 10 validation and training sets from $\mathbf{Q}^{f,\,s}$ were used to produce 10 classifiers $C_{i,j}^{\text{window}}$ using the optimal parameters, $\zeta^{*f,s}$. Each fold was created so that there was an equal number of instances from both face feature and non-face feature classes.[2]

[2] Known as stratified k-fold cross-validation.

2.3.3 Localization performance evaluation

The performance of the face feature localization system was measured using the percentage area overlap ρ of the predicted face feature window and the ground truth face feature window as determined by the manual annotation process.

2.4 Semantic filtering

Similar to the lower-bounding lemma concept in information retrieval, our semantic filtering module tries to discard images from the database that are unlikely to be related to the query image on a semantic level. To remove semantically irrelevant images, we apply the LPM as a semantic filter by comparing each database image with the query image. The following section outlines the LPM concept followed by the semantic filter process.

2.4.1 Learning pseudo metric

The LPM concept was proposed in [11] and applied to resolving semantic data classification and clustering problems [21]. Indeed, the LPM is an implementation of characteristic function of equivalent class of data. From our previous studies with simulation results, the LPM is capable of serving as a filter for removing irrelevant images at the semantic level. In this paper, the LPM is employed to classify whether any two points in a feature space χ belong to the same class or not [11]. The following briefly reviews some definitions related to the pseudo metric and the LPM concept.

Definition 2.1 Let χ be a set of points and d be a real-valued function defined over the set $\chi \times \chi$. The function d is called a pseudo metric on χ if it satisfies the following three axioms for all $x, y, z \in \chi$.

1. $d(x, x) = 0$ (the axiom of reflexivity)
2. $d(x, y) = d(y, x)$ (the axiom of symmetry)
3. $d(x, z) \leq d(x, y) + d(y, z)$ (the axiom of triangle inequality).

As the equivalence relation is commonly used to group together objects that share a common similarity we can define;

Definition 2.2 An equivalence relation, denoted by $\sim$, on a set χ is a binary relation on χ that is reflexive, symmetric and transitive for all $x, y, z \in \chi$.

Definition 2.3 Given a set χ and an equivalence relation $\sim$ over χ, an equivalence class is a subset of χ of the form $\{x \in \chi | x \sim a\}$ where a is an element in χ. This equivalence class is usually denoted as $[a]$; it consists of precisely those elements of χ which are equivalent to a.

The pseudo metric can be defined via the use of equivalence classes such that given a set of χ and a set of equivalence classes, $\{[a_j], j = 1, 2, \ldots, p\}$, which are derived from an equivalence relation $\sim$ over χ.

Proposition 2.1 *The function f defined by* (2) *is a pseudo metric over* χ.

$$f(x,y) = \begin{cases} 1 & \text{if } x \in [a_i] \text{ and } y \in [a_j], i \neq j, \\ 0 & \text{if } x \in [a_i] \text{ and } y \in [a_j], i = j. \end{cases} \quad (2)$$

The LPM is an implementation of the characteristic function f through learning techniques. To verify the quality of approximation of the function f by leaner models, we proposed the following practical criteria (PC) [11]:

1. $Y_{NN}(x,x) \leq \epsilon_1$ as x and y belong to the same class.
2. $Y_{NN}(x,y) \geq \epsilon_2$ as x and y belong to different classes.
3. $|Y_{NN}(x,y) - Y_{NN}(y,x)| \leq \epsilon_3$ for any x and y.
4. $Y_{NN}(x, z) \leq Y_{NN}(x, y) + Y_{NN}(y, z)$ as x and y belong to different classes.

Here, Y_{NN} represents the neural network output, ϵ_1 and ϵ_3 take smaller values (e.g. 0.2–0.3), ϵ_2 takes a larger value (e.g. 0.7–0.8), which implies a higher probability of the axiom of triangle inequality holding. For more details on the learning process, readers may refer to our previous work in [11, 21].

2.4.2 Filter design

To perform semantic filter on the database images, we use the LPM to compare the query image with all other images in the database. This is done by combining the feature vectors from each face feature into a single feature vector, V. The database face and query face feature vectors are passed to the LPM, and the resulting output $Y(X_i, X_q)$ is produced. If $Y(X_i, X_q)$ is below the threshold, λ, then the database image i is deemed to be semantically similar to the query image and is retained for the second stage of retrieval. Otherwise, the image is discarded. After performing semantic filtering, the resulting subset of database images are compared with the query image using the Euclidean distance measure. The database images are then ranked according to the distance scores.

2.5 Face feature-based retrieval

Unlike most FIR systems that use features extracted from the whole face, we have approached the problem using features derived from each face feature. The face feature localization system produces a set of feature vectors, $\mathbf{X}_q = [\mathbf{X}_q^{\text{left eye}}, \mathbf{X}_q^{\text{right eye}}, \mathbf{X}_q^{\text{nose}}, \mathbf{X}_q^{\text{mouth}}]$, which describe each face feature of a query image $\mathbf{I}_q^{\text{face}}$. The query feature vectors are compared with each face image in the database (after semantic filtering), $\mathbf{X}_i = [\mathbf{X}_i^{\text{left eye}}, \mathbf{X}_i^{\text{right eye}}, \mathbf{X}_i^{\text{nose}}, \mathbf{X}_i^{\text{mouth}}]$, and the used similarity measure $S(\mathbf{X_i}, \mathbf{X_q})$ is defined as:

$$S(\mathbf{X_q}, \mathbf{X_i}) = \sum w^f \|\mathbf{X}_q^f - \mathbf{X}_i^f\| \quad (3)$$

where w^f is a face feature weight subject to $\sum w^f = 1$ and is set by the users. The database images are then ranked in ascending order according to $S(\mathbf{X_i}, \mathbf{X_q})$, and the top 12 images are shown to the users.

2.6 Retrieval performance metrics

2.6.1 Mean rank

The FIR system performance is measured using the average mean rank, μ (4), and overall precision, p (6).

$$\mu = \sum_{q=1}^{Q} \mu_q, \quad (4)$$

where Q is the number of query images, μ_q is the mean rank for a given query q and is defined as:

$$\mu_q = \frac{N_q(N_q + 1)}{2\sum_{i=1}^{N_q} r_{q,i}} \quad (5)$$

where N_q is the number of relevant images in the database for query image q, $r_{q,i}$ is the rank of relevant database image i in the returned results. μ_q is sensitive to the ranking order of the relevant images. The higher the rank of relevant images from the database, the closer μ_q is to 1. While the lower the rankings of relevant retrieved images, μ_q approaches 0.

2.6.2 Precision

The overall precision, p, is the percentile of database images that are relevant with respect to the query image in the first 12 retrieved images. Where p_q is the precision for a query image q.

$$p = \frac{1}{Q}\sum_{q=1}^{Q} p_q, \quad (6)$$

where p_q is defined as:

$$p_q = \frac{1}{12}\left(\sum_{i=1, r_{q,i} \leq 12, T_{q,i}=0}^{1379} 1\right) \quad (7)$$

3 Performance evaluation

The following section details how the proposed semantic filtering-based FIR system is evaluated.

Table 1 Number of instances of cropped and normalized face images

Gender	No. people	AR^{d}_{Train}		AR^{d}_{Test}	
		Images per person	Total	Images per person	Total
Male	63	8	504	4	252
Female	52		416		208
Total	115		920		460

3.1 Experiment setup

3.1.1 Face image dataset

In this study, we used the AR dataset [22] that consists of photographs of people taken over two sessions, two weeks apart. The first session had 135 people; then, two weeks later, 120 people from the first session were photographed again. There was no restriction on individuals to keep their hairstyle, cloths, or make-up the same between sessions. During both first session and second session shoots, each person was photographed 13 times with different expressions, lighting conditions, and with/without sunglasses or scarf.[3]

As our system does not yet handle faces images with extreme facial expressions and occlusions, we excluded images with such characteristics from the dataset. This left 6 images per person per session. The remaining images from both sessions are combined and called *AR*. The *AR* dataset was randomly split into a training and test sets, denoted AR_{Train} and AR_{Test}, respectively. Table 1 lists the total number of images for each session.

The AR images were 768 × 576 in 24-bit color but were converted to 8-bit gray-scale using $Y = 0.299R + 0.587G + 0.114B$, where R, G, and B are the red, green, and blue values of each pixel, respectively, and Y is the resulting gray value. Because the faces vary in scale, we scaled the images so the interpupil distance, denote d, was 20 and 40 pixels, producing separate datasets. This yields AR datasets AR^{d}_{Train} and AR^{d}_{Test}.

Three datasets need to be created for training the face feature localization classifier, training the LPM classifier, and finally for FIR performance evaluation. First, to train the localization classifier, we create a training set from manually selected face feature images (ground truth face feature images). After training the localization classifier, we automatically extract face features from both AR^{d}_{Train} and AR^{d}_{Train}. Feature extraction is applied to each face feature images to yield four feature vectors $X_i^{f,\,s}$. The four extracted feature vectors are compiled into a single vector X_i^{d}, which represents the content of the face image i.

[3] The 13 images consisted of neutral expression, smiling, angry, screaming, neural expression with right light, neural expression with left light, neural expression with both lights on, sunglasses, sunglasses with right light on, sunglasses with left light on, scarf, scarf with right light on, and scarf with left light on.

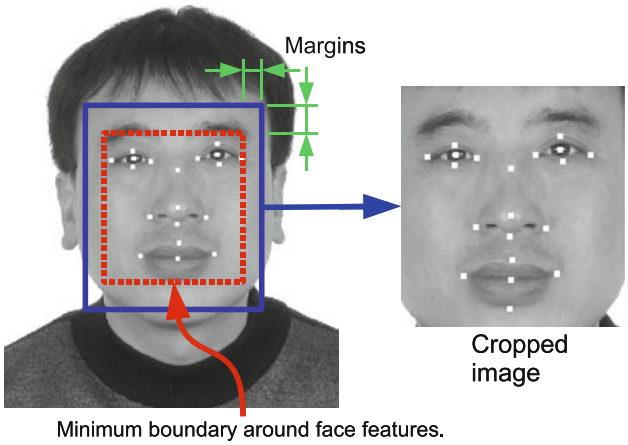


Fig. 8 Example of face annotation and the automatically cropped face

3.1.2 Data preprocessing

Localization data preparation We generated the training and validation sets of feature vectors for localization training from AR^{d}_{Train} as detailed in Sect. 2.3.1. The AR^{d}_{Test} face image set is used to test the face feature localization accuracy of the trained learners.

First, we manually annotated each face in the dataset, marking the center of the eye pupils, tip of the nose, which can be used for and the center of the mouth. The boundaries of each face feature were also marked by four points, see Fig. 8 for an example. The reason for manually marking face features are twofold. The first reason is that it allows us to automatically crop the faces so that they are well framed. The second reason is that it allows us to consistently generate a set of ground truth face feature images, which can be used for training the classifier as well as it provides a means to measure localization performance of our system.

Cropping a large number of images by hand would be cumbersome; hence for each face image, we find a minimum rectangular boundary to encapsulate the face features as marked by the annotation process. We then add a margin to the minimum boundary to produce the cropped face image. Figure 8 illustrates the manually annotated face image and resulting cropped face image as well as the annotation points for each face feature.

The images are then histogram normalized to reduce the effect of illumination variation between the different images. Next, all images were rotated to ensure the eye-pupil centers aligned horizontally, removing the in-plane rotations.

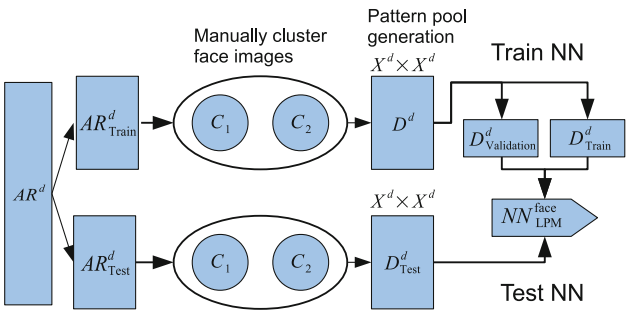


Fig. 9 Overview of LPM training

LPM data preparation To train the LPM, we created a pattern pool as described in Sect. 2.4.1. We manually clustered the face images in AR_{Train} and AR_{Test} into two groups according to gender. The pattern pools D^d and $D^{\text{window}}_{i,j}$ are created using the Cartesian operator applied to the combined local face feature vectors $X^d_i = [X^{\text{left eye},s,d}_i, X^{\text{right eye},s,d}_i, X^{\text{nose},s,d}_i, X^{\text{mouth},s,d}_i]$. D^d was randomly split into D^d_{Train} and $D^d_{\text{Validation}}$ for building the LPM model. Figure 9 shows the overview of the LPM data preparation and training.

FIR data preparation Finally, to evaluate the FIR system, we used all face images from the same individuals that appear in both the first and second sessions. For each selected image, the semantic filtering is applied using the combined local feature vector. The database images that pass the filter stage are then compared with the query image using Euclidean distance.

3.2 Neural network settings

3.2.1 Localization settings

We used a three layer, fully connected feed-forward neural network with a single output neuron for face feature localization. All neurons used the logarithmic sigmoid activation function. The network weights were initialised using the Nguyen and Widrows algorithm and trained using the scaled conjugate gradient algorithm [23]. The parameters we aimed to optimize were the number of hidden neurons, $\zeta_{NN} = h_{NN}$ using the hold-out dataset with 80% for training and 20% as a validation set. Training of each neural network was continued until the validation root mean square error started to increase after reached a minimum, training was stopped, and the network topology with the highest validation classification accuracy was selected.

3.2.2 LPM settings

For the neural network-based LPM classifier, we again used a three layer, fully connected feed-forward neural network with a single output neuron. All neurons used the logarithmic sigmoid activation function. The network weights were initialised using the Nguyen and Widrows algorithm and were trained using the scaled conjugate gradient algorithm [23].

The pattern pool D^d was randomly split 60%/40% to create the $D^{\text{window}}_{i,j}$ and $D^d_{\text{Validation}}$ datasets. Training of each neural network topology was continued until the validation root mean square error started to increase after reached a minimum, training was stopped, and the network topology with the highest validation classification accuracy was selected.

4 Results and discussion

4.1 Experiment results

4.1.1 Neural network localization performance

The number of hidden neurons and s that produced the highest validation classification accuracy for each face feature and scale d were selected during training. Tables 2 and 3 present the training and test localization accuracy for image sets AR^d_{Train} and AR^d_{Test} with an eye distances of $d = 20$ and $d = 40$ pixels, respectively, where L denotes the left eye, R the right eye, N the nose, and M the mouth, respectively.

4.1.2 LPM training performance

Table 4 presents the LPM performance as measured by the practical criteria (Sect. 2.4.1) for image sets

Table 2 NN tenfold cross-validation localization performance for each face feature from AR^{20}_{Train} and AR^{20}_{Test}

f	s	h^*_{NN}	AR^{20}_{Train}		AR^{20}_{Test}	
			ρ	SD	ρ	SD
L	Small	25	76.52	2.42	77.11	2.85
R	Small	25	74.91	2.57	77.72	3.66
N	Large	25	82.61	1.03	81.66	1.56
M	Ave	25	73.67	2.11	75.47	1.60

Table 3 NN tenfold cross-validation localization performance for each face feature from AR^{40}_{Train} and AR^{40}_{Test}

f	s	h^*_{NN}	AR^{40}_{Train}		AR^{40}_{Test}	
			ρ	SD	ρ	SD
L	Small	30	80.02	1.52	78.65	1.75
R	Small	30	79.10	1.81	79.64	1.48
N	Large	30	78.11	1.46	78.83	1.64
M	Large	30	71.90	2.19	72.60	1.45

Table 4 LPM performance according to the four LPM practical criteria on the pattern pool data where $\epsilon_1 = \epsilon_3 = 0.3$ and $\epsilon_2 = 0.7$

d	h_{LPM}	PC_1	PC_2	PC_3	PC_4
D^d_{train}					
20	60	0.8349	0.8327	0.8195	0.7065
40	100	0.8162	0.8203	0.8384	0.7440
$D^d_{\text{Validation}}$					
20	60	0.7670	0.7653	0.8052	0.6203
40	100	0.7711	0.7749	0.8260	0.7908
D^d_{test}					
20	60	0.6543	0.6492	0.7440	0.8533
40	100	0.6759	0.6911	0.8072	0.6818

D^d_{train}, $D^d_{\text{validation}}$, and D^d_{test} with eye distances of $d = 20$ and $d = 40$ pixels, respectively. The results are lower compared with the previous applications of the LPM within a k-NN classifier for semantic image classification [11]. However, given the face images are not easily grouped into distinct clusters for creation of the pattern pool, it is not unexpected that the training performance of the LPM is lower in our system.

4.1.3 FIR performance

Table 5 presents the performance of the FIR system without filtering the database images using the LPM. Table 6 shows the retrieval performance, with standard deviation, of our local face feature-based FIR system with semantic filtering for $\lambda = 0.3$. Where "filtered db size" is the number of images in the filtered database, μ-rank is the mean rank, and p is the average precision.

Comparing the filtered and unfiltered FIR results, it can be seen that the LPM filtering method removed on average 50% of the images from the original database without reducing the performance. On the surface, the average mean rank and precision improved for the filtered database. However, the standard deviation increased by a factor of 4 to 5, indicating that the stability of the results can vary between query images.

As the LPM is trained on pattern pools derived from the face images grouped according to gender, we initially expected poorer FIR performance as we did not believe that face features alone would capture sufficient information about the face to differentiate between genders. However, the results show otherwise, and indicate that demographic information, such as gender, can be captured using relatively small parts of the face image, rather than the whole face image.

Because the performance measured by the μ-rank and precision is very low compared with other FIR systems, we present a number of examples of the retrieved images for both unfiltered and filtered databases to show the actual potential of the system.

Figure 10a shows the top 12 ranked images for face image w-058 with a neutral expression wearing corrective eye wear. Of the 12 retrieved images, the query image is at the top center and the retrieved images are ranked from one to 12 left to right, top to bottom. The image class label and distance values are also presented. Four of the retrieved faces are also wearing corrective eye wear and eight faces have large/thick lips similar to the query face. The majority of retrieved faces are of the same sex, indicating that the system is able to retrieve images with similar demographics.

While the retrieved faces for the same query using LPM semantic filtering are shown in Fig. 10b, It shows that a number of the top ranked images were removed during the LPM filtering stage. However, two database faces wearing glasses were ranked higher.

Table 5 Face image retrieval performance with no filtering ($\lambda = 1.0$)

Data	No filtering		
	db size	μ-rank	p
AR^{20}	1379	0.0155 ± 0.0094	0.0981 ± 0.0866
AR^{40}	1379	0.0163 ± 0.0134	0.0954 ± 0.0945

Table 6 Face image retrieval performance using semantic filtering ($\lambda = 0.3$)

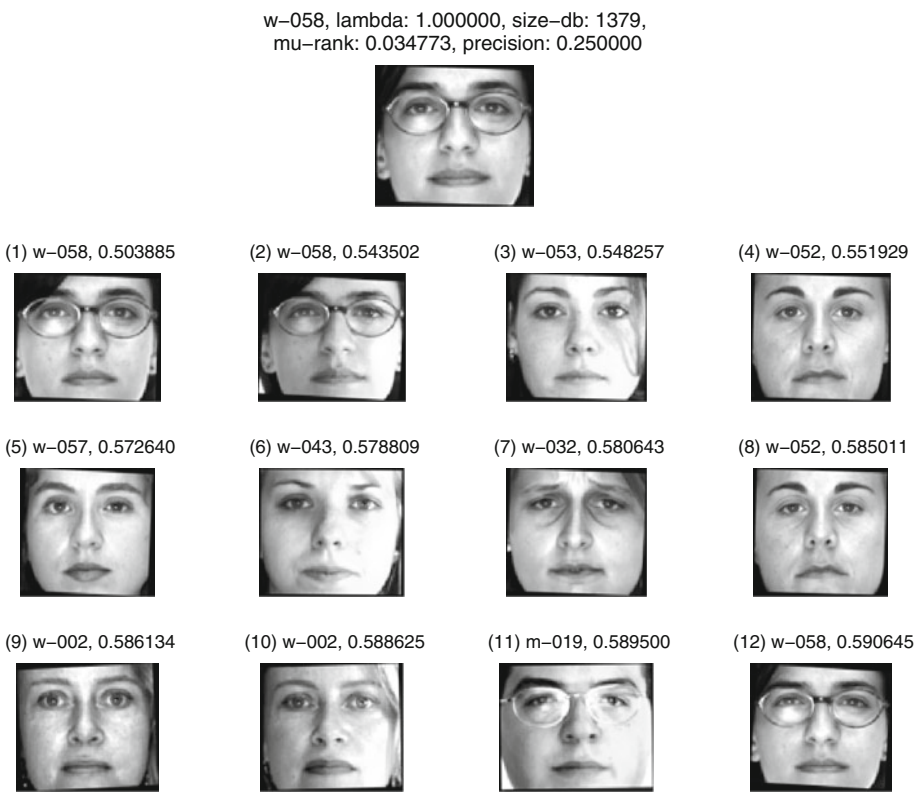
Data	Semantic filtering		
	Filtered db size	μ-rank	p
AR^{20}	593 ± 122	0.0263 ± 0.0313	0.0885 ± 0.0884
AR^{40}	612 ± 116	0.0309 ± 0.0612	0.0908 ± 0.0940

4.2 Robustness analysis

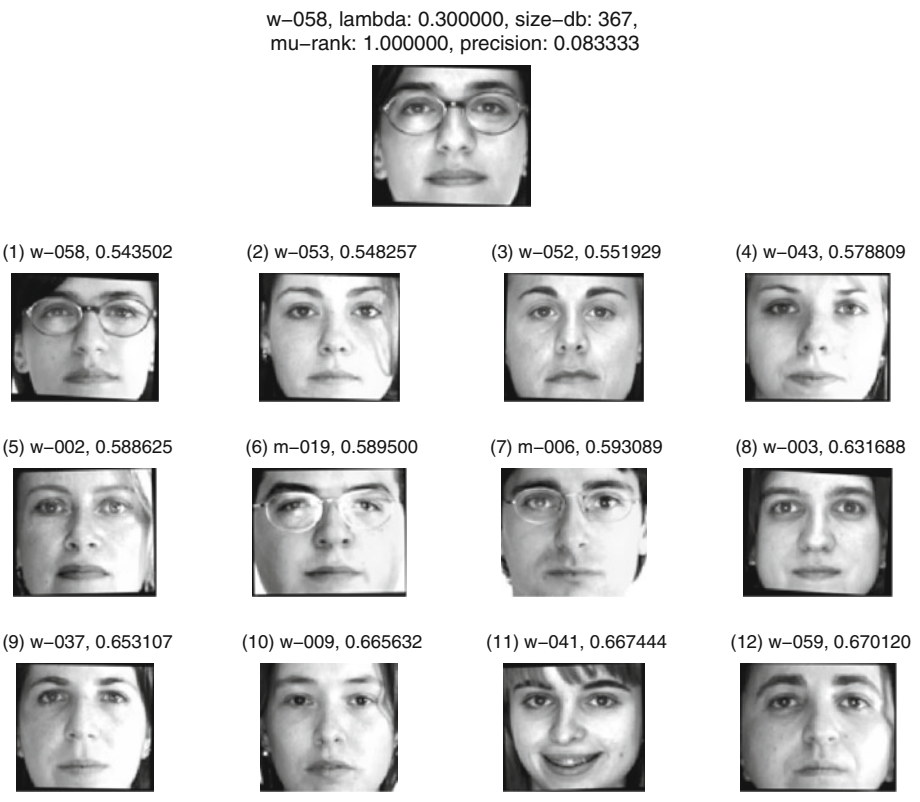
The robustness of the FIR system with the LPM filtering step was examined by varying the λ threshold over [0, 1]. Figure 11a, b, c, and d show the effect of varying λ on the filtered database size, the number of relevant images in the database N_q, μ-rank and precision, respectively, for $d = 40$. Each graph is plotted with error bars (standard deviation).

There is an initial large drop in the size of the filtered database for $\lambda = 0.9$, then a steady linear decline in database size until $\lambda = 0.1$, Fig. 11a. Most of the relevant images from the same person are present in the filtered database, with two to three images from the same class being filtered out by the LPM classifier, Fig. 11b.

The average mean rank and precision performance is very stable of almost the entire λ threshold range, as seen in

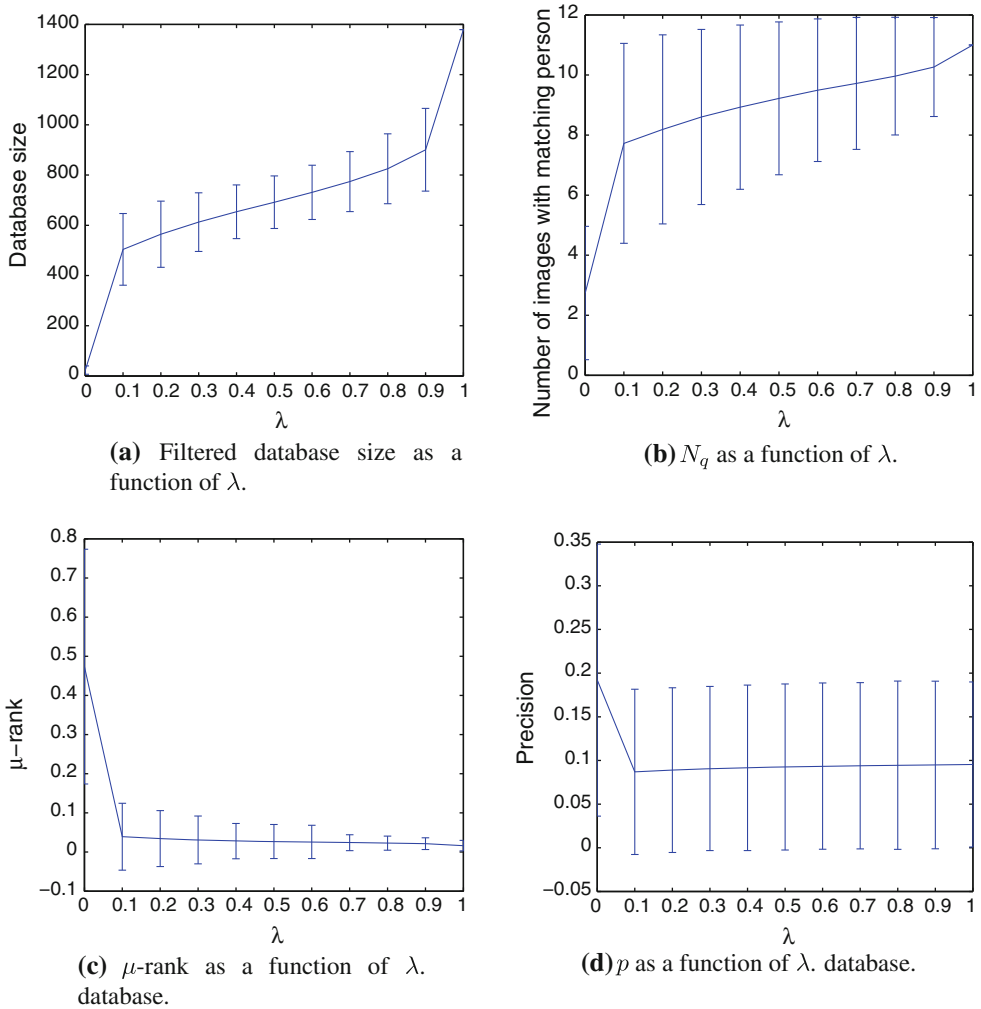


(a) Top 12 images from un-filtered database.



(b) Top 12 images from filtered ($\lambda = 0.3$) database.

Fig. 10 Top 12 retrieved images for "W-058" from filtered and unfiltered image database



(a) Filtered database size as a function of λ.

(b) N_q as a function of λ.

(c) μ-rank as a function of λ. database.

(d) p as a function of λ. database.

Fig. 11 Robustness of the FIR system with respect to the λ

Fig. 11c and d. However, the standard deviation increases as the threshold reduces and more database images are filtered, producing more erratic results for any given query image. Such that, the returned results for a query image are unaffected by the filter stage, and in other cases, the return images are quite different.

5 Conclusions

Our proposed face image retrieval (FIR) system performs poorly according to the mean rank and precision metrics. However, the μ-rank only measures how well faces of the same class (same person) have been ranked, whilst examining the images retrieved for the query images shows that the retrieved images (which are of different people) do indeed appear similar to the query face. There are no apparent alternative metrics for measuring FIR performance due to the subjective nature of similarity between faces.

We found that our face feature localization system performs sufficiently well to localize the face features for the purpose of FIR. Also, the neural network-based LPM was found to be very effective at removing semantically irrelevant (by gender) images without reducing the FIR performance significantly. Our initial expectation was that the semantic filtering would not perform well as local face features would not be sufficiently rich enough to capture the difference between genders. However, the results demonstrated this not to be the case.

Face image retrieval is a challenging task and has not received sufficient attention. Technically, this problem is closely linked to face recognition; however, they differ in terms of query format. FIR system design depends on the

format of query, which can be of a sample image, sketch, or even verbal description. Further research on this topic moves around system design with various query formats and domain applications.

References

1. Datta R, Joshi D, Li J, Wang JZ (2008) Image retrieval: ideas, influences, and trends of the new age. ACM Comput Surv 40:1–60
2. Fauzi MFA, Lewis PH (2009) Query by low-quality image. Image Vis Comput 27:713–724
3. Liu W, Tang X, Liu J (2007) Bayesian tensor inference for sketch-based facial photo hallucination. In: Veloso MM (ed) Proceeding international joint conference of artifical intelligence. Morgan Kaufmann Publishers Inc., San Francisco, pp 2141–2146
4. Sridharan K, Nayak S, Chikkerur S, Govindaraju V (2005) A probabilistic approach to semantic face retrieval system. In: Kanade T, Jain A, Ratha NK (eds) Audio- and video-based biometric person authentication, vol 3546 of lecture notes in computer science. Springer, Berlin, pp 977–986
5. Ai H, Liang L, Xiao X, Xu G (2001) Face indexing and retrieval in personal digital album. In: Proceedings of 2nd IEEE Pacific Rim conference multimedia, vol 2195, Springer-Verlag, London, UK, pp 48–54
6. Gao Y, Qi Y (2005) Robust visual similarity retrieval in single model face databases. Pattern Recognit 38:1009–1020
7. Wu B, Ai H, Huang C (2004) Facial image retrieval based on demographic classification. In: Proceedings of 17th International Conference of Pattern Recognition, vol 3. IEEE Computer Society, Washington, pp 914–917
8. Santini S, Jain R (1999) Similarity measures. IEEE Trans Pattern Anal Mach Intell 21:871–883
9. Smeulders AWM, Worring M, Santini S, Gupta A, Jain R (2000) Content-based image retrieval at the end of the early years. IEEE Trans. Pattern Anal Mach sIntell 22:1349–1380
10. Liu Y, Zhang D, Lu G, Ma W-Y (2007) A survey of content-based image retrieval with high-level semantics. Pattern Recognit 40:262–282
11. Wang DH, Ma XH, Kim YS (2005) Learning pseudo metric for intelligent multimedia data classification and retrieval. J Intell Manuf 16:575–586
12. Turk M, Pentland A (1991) Eigenfaces for recognition. Cogn Neurosci 3:71–86
13. Belhumeur PN, Hespanha J, Kriegman DJ (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE Trans Pattern Anal Mach Intell 19:711–720
14. Cai D, He X, Han J, Zhang H-J (2006) Orthogonal laplacianfaces for face recognition. IEEE Trans Image Process 15:3608–3614
15. He X, Cai D, Han J (2008) Learning a maximum margin subspace for image retrieval. IEEE Trans Knowl Data Eng 20:189–201
16. Yang M-H, Kriegman DJ, Ahuja N (2002) Detecting faces in images: a survey. IEEE Trans Pattern Anal Mach Intell 24:34–58
17. Zhou H, Yuan Y, Sadka AH (2008) Application of semantic features in face recognition. Pattern Recogn 41:3251–3256
18. Conilione P, Wang DH (2011) Automatic localization and annotation of facial features using machine learning techniques. Soft Comput 15:1231–1245
19. Wang DH, Ma XH (2005) A hybrid image retrieval system with user's relevance feedback using neurocomputing. Informatica 29:271–279
20. Lu Y, Guo H, Feldkamp L, Robust neural learning from unbalanced data samples. In: Proceedings of IEEE International Joint Conferences of Neural Networks, vol 3. Anchorage, Alaska, USA, pp 1816–1821
21. Wang DH, Kim Y-S, Park SC, Lee CS, Han YK (2007) Learning based neural similarity metrics for multimedia data mining. Soft Comput 11:335–340
22. Martinez AM, Benavente R (1998) The AR face database, technical report 24, Computer Visual Center
23. Mller MF (1993) A scaled conjugate gradient algorithm for fast supervised learning. Neural Netw 6:525–533