Where Is Who: Large-Scale Photo Retrieval by Facial Attributes and Canvas Layout

Yu-Heng Lei, Yan-Ying Chen, Bor-Chun Chen, Lime Iida, Winston H. Hsu
National Taiwan University, Taipei, Taiwan
{ryanlei, yanying}@cmlab.csie.ntu.edu.tw,
{siriushpa, limeiida}@gmail.com, winston@csie.ntu.edu.tw

ABSTRACT

The ubiquitous availability of digital cameras has made it easier than ever to capture moments of life, especially the ones accompanied with friends and family. It is generally believed that most family photos are with faces that are sparsely tagged. Therefore, a better solution to manage and search in the tremendously growing personal or group photos is highly anticipated. In this paper, we propose a novel way to search for face photos by simultaneously considering attributes (e.g., gender, age, and race), positions, and sizes of the target faces. To better match the content and layout of the multiple faces in mind, our system allows the user to graphically specify the face positions and sizes on a query "canvas," where each attribute combination is defined as an icon for easier representation. As a secondary feature, the user can even place specific faces from the previous search results for appearance-based retrieval. The scenario has been realized on a tablet device with an intuitive touch interface. Experimenting with a large-scale Flickr[1] dataset of more than 200k faces, the proposed formulation and joint ranking have made us achieve a hit rate of 0.420 at rank 100, significantly improving from 0.036 of the prior search scheme using attributes alone. We have also achieved an average running time of 0.0558 second by the proposed block-based indexing approach.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query formulation; H.5.2 [**User Interfaces**]: Input devices and strategies

[1]All of the face images presented in this paper except for those by Google Image Search in Fig. 2 (b) and Fig. 4 attribute to various Flickr users under a Creative Commons License.

SIGIR'12, August 12–16, 2012, Portland, Oregon, USA.

Figure 1: Example queries and top 5 retrieval results from our photo search system. (a) specifies two arbitrary faces with the larger one on the left and the smaller one on the right. (b) further constrains that the left face has attributes "female" and "youth" and the right face has attribute "kid." (c) specifies two faces of "male" and "African" on the left and right, in addition to an arbitrary face on the center. (d) specifies a particular face in the database at the desired position and in the desired size. (e) specifies the previous database face on the left, and a face of "female" and "youth" on the right.

Keywords

Face attributes, Face retrieval, Touch-based user interface, Block-based indexing

1. INTRODUCTION

The ubiquitous availability of digital cameras has made it easier than ever to capture moments of life, especially the ones accompanied with friends and family. It is generally believed that most family photos are with faces that are sparsely tagged. Therefore, a better solution to manage and search in the tremendously growing personal or group photos is highly anticipated.

Psychology research in perception shows that images with certain kinds of subjects attract more attention of the eyes [8]. Among these subjects, human faces are the most memorable, followed by images of human-scale space and close-ups of objects [11]. The phenomena becomes more obvious in consumer photos because most of them contain family

Figure 2: Illustration of searching for a face image that the user remembers. The search intention is indicated in the cloud icon in (a), where there are a boy's face on the left and a larger girl's face on the top right of it. Four types of approaches are shown. (b) is Google's text-based image search with advanced options of searching only face images. (c) is facial-attribute-based image search with text-based queries, similar to the scheme proposed by [15]. (d) is image search based on face positions and face sizes. (e) is performed by simultaneously considering facial attributes, face positions, and face sizes. Images squared in green solid lines (blue dashed lines) are believed to be highly (partially) relevant by an average user. The results in (e) best match the search intention, showing the power of multimodal fusion in retrieval systems.

members or close friends that the user cares about and usually keeps in mind. Therefore, they are able to make use of the face content and the face layout that they remember to effectively formulate their search intentions. Furthermore, viewing the retrieved images probably recalls more scenes in the user's memory, so they expect to be able to refine their query interactively. For example, "viewing a photo of Alice standing next to me, it reminds me of another photo with an African kid sitting in the middle of us." Although consumer photos generally lack annotations, automatic face analysis techniques would make the scenario economical and scalable.

In this paper, we propose a novel system for searching consumer photos by automatically analyzing "wild photos" (without tag information at all) through facial attribute detection (Sec. 4.1) and appearance similarity estimation (Sec. 4.2). To better match the content and layout of the multiple faces in mind, rather than laboriously sketching detailed outline or typing text, our system allows the user to graphically specify the face positions and sizes on a query "canvas," where each attribute combination is defined as an icon for easier representation. The query can be simply finding arbitrary faces in the desired layout (Fig. 1 (a)), or further constrained by facial attributes (Fig. 1 (b) and (c)). As a secondary feature, the user can even place specific faces from the previous search results for appearance-based retrieval (Fig. 1 (d)), combined with other attributed faces (Fig. 1 (e)). Other complicated search intentions also apply.

The scenario has been realized on a tablet device with an intuitive touch interface where the user can easily refine their query by interacting with the real-time search results. To provide effective matching in a large-scale Flickr dataset of more than 200k faces, the proposed formulation and joint ranking have made us achieve a hit rate of 0.420 at rank 100, significantly improving from 0.036 of the search scheme proposed by [15] using attributes alone. To provide efficient retrieval, we have also achieved an average running time of 0.0558 second by the proposed block-based indexing approach. The numbers are scalable to even larger photo collections.

The contributions of this paper are as follows:

- Propose the problem in how to formulate search intentions for face images as tangible search queries, i.e., by graphically specifying face content and layout on a query "canvas." We also provide an intuitive touch-based interface for refining the search results interactively.

- Propose a formulation for matching multiple faces between the query canvas and the target image (Sec. 5.1) and effectively match a single face by simultaneously considering attributes, appearances, positions, and sizes (Sec. 5.2).

- Propose a block-based indexing approach for efficient retrieval (Sec. 5.4).

2. OBSERVATIONS AND RELATED WORK

In this section, we review various query formulations and query modalities in image search systems and their applicabilities to face photo search. Fig. 2 is an illustration of such a scenario. The target image in the user's mind (Fig. 2 (a)) is a boy's face on the left and a larger girl's face on the top right of it. The user vaguely remembers the face content and layout, but not the exact image file in the collection.

Existing commercial image search engines mostly rely on matching the query keywords with the surrounding text or manual tags of the target images. Fig. 2 (b) is obtained by Google Image Search using the keywords "boy girl" with advanced options of searching only face images. Directly matching text not only reveals little about the image content, but in this particular case, it also happens to match the movie title "It's a Boy Girl Thing" and retrieves some

irrelevant images in the scene. What's worse, tags are often inaccurate, incorrect, or ambiguous [12]. Due to the complex motivations behind tag usage [2], tags do not necessarily describe the content of an image [13].

In content-based image retrieval, Kumar [15] proposes facial attribute classification by SVM and AdaBoost, and uses the confidence scores for image retrieval. Fig. 2 (c) is produced in a similar way by enabling only the attribute modality in our system. The corresponding attributes specified are "male + kid" for boy and "female + kid" for girl. While the attributes (especially the age) are mostly correct, this approach does not consider the face layout in the user's mind at all. On the other hand, Fig. 2 (d) is produced by enabling only the position and size modalities in our system. While the face layouts are highly relevant due to accurate face detection, this approach does not consider about the face content. To utilize both the content and layout information, Fig. 2 (e) is produced by the full version of our system that combines all of these three modalities. The results in Fig. 2 (e) best match the user's search intention in terms of finding highly relevant (squared in green solid lines) and partially relevant (squared in blue dashed lines) images. The above illustration shows the power of multimodal fusion in retrieval systems.

Some efforts also attempt to capture the user's search intention by visually describing both the image content and layout on a query canvas. Thanks to the growing popularity of touch devices, it has become more intuitive and convenient than ever to formulate such queries. [3] revisits the problem of sketch-based image search for scene photos. However, the gap between the user's mind and their specified query can still be large even in such a system. For instance, users with poor drawing skills may have a hard time describing their intention accurately. In addition, some object details are naturally difficult to sketch, and many concepts are even more difficult to describe by sketching, such as the age of a face. Therefore, the practicability of sketch-based retrieval for photo management is questionable, especially for face photos.

To deal with this sketching difficulty, [19] allows the user to formulate a 2-D "semantic map" by placing text boxes of various search concepts at desired positions and in desired sizes. However, it is intended for generic objects, not for faces of different individuals. To apply to face photo management, [14] also allows the user to specify face positions, and face sizes on a canvas. These faces are further described by tagging names and even drawing social relationships [17]. However, non of these efforts proposes an efficient indexing method for large-scale photo retrieval. Meanwhile, typing text is not the most intuitive operation on touch devices even though these efforts aim for better user experience.

Specifically for photo management, some commercial services (e.g., Picasa [18] and iPhoto [10]) that exploit face recognition technologies to help face annotation in a semi-supervised or supervised manner have been shown promising. However, people are mostly reluctant to annotate their photos, especially when photos are taken enormously due to the ubiquitous availability of digital cameras. Also, many of consumer photos are group photos, which makes the face annotation task even more tedious. In our work, we further consider spatial layout, attributes, and appearance for face photo retrieval. We believe it can be complementary[2] to existing face annotation solutions.

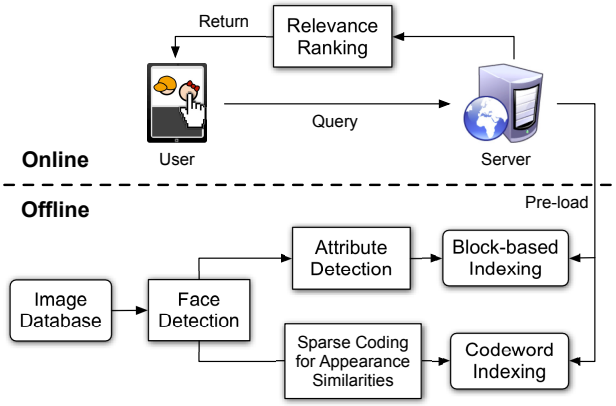

Figure 3: An overview of our proposed system. Photos are analyzed offline through face detection, facial attribute detection, and sparse coding for appearance similarities. The results are incorporated into the proposed block-based index and codeword index for efficient retrieval.

3. SYSTEM OVERVIEW

Fig. 3 is an overview of our proposed system named "Where is Who." In the offline process, the image database first goes through face detection to identify and locate frontal faces in the images. These faces are then analyzed through facial attribute detection (Sec. 4.1) and sparse coding for appearance similarities (Sec. 4.2). Finally, the attribute scores along with the position and size information are incorporated into the block-based index (Sec. 5.4). The sparse codes of faces are also stored in the codeword index. These indices are pre-loaded for rapid online response. In the online process, the server retrieves candidate images in inverted lists, ranks them by relevance (Sec. 5.2), and returns the search results back to the user. Note that appearance-based retrieval is treated as a secondary feature and is not evaluated throughout this paper.

4. IMAGE ANALYSIS

4.1 Detecting Facial Attributes

Facial attributes possess rich information about people and have been shown promising for seeking specific persons

[2]Face recognition or face annotation information can be exploited as another source of the "face content" considered in this work.

Table 1: The 3 attribute types and 8 corresponding attributes detected in our system.

Type	Attribute
Gender	male, female
Age	kid, youth, elder
Race	Caucasian, Asian, African

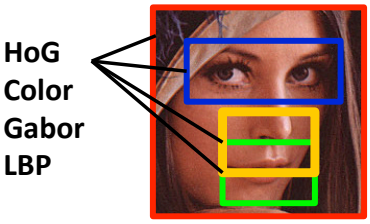

Figure 4: For each of the four face components (whole face, eyes, nose, and mouth), four low-level features (HoG, grid color moments, Gabor, and LBP) are extracted. Each of 16 combinations (e.g., <mouth, LBP>) is treated as a mid-level feature for which an SVM is learned.

in face retrieval and surveillance systems. In this work, we utilize 8 facial attributes (Table 1) including 2 of gender (male, female), 3 of age (kid, youth, elder) and 3 of race (Caucasian, Asian, African) to profile faces in large-scale photos.

In the training phase, each attribute classifier is learned separately through a combination of Support Vector Machines (SVMs) and Adaboost [9] similar to [15]. Firstly, we crawl user-contributed photos from Flickr and extract facial regions by a face detector. The face images are annotated manually with positive and negative class labels. As illustrated in Fig. 4, the faces are then automatically decomposed into four different face components, i.e., whole face, eyes, nose, and mouth. From each of these components, four low-level features, i.e., histogram of oriented gradients (HoG) [6], grid color moments, Gabor filter, and local binary patterns (LBP) [1] are extracted.

A mid-level feature learned is an SVM with a specific low-level feature extracted from a specific face component, e.g., an SVM for <mouth, LBP>. Finally, the optimal weighting of the 16 (4×4) mid-level features for this attribute is determined through Adaboost. The combined strong classifier represents the most important parts of that attribute. For example, <whole face, Gabor> is most effective for the female attribute while <whole face, color> is most effective for the African attribute.

Experimenting with the benchmark data [15], the approach can effectively detect facial attributes and achieve an accuracy of more than 80% on average. Meanwhile, the training framework is generic for various cases thus providing a potential to extend to more attributes[3].

4.2 Sparse Coding for Appearance Similarities

To enable search through face appearance, we adopt the face retrieval framework of [5]. The advantage of this framework includes: (1) efficiency, which is achieved by using sparse representations of face image with inverted indexing, and (2) leveraging identity information, which is done by incorporating the partially-tagged identity information into the optimization process of codebook construction. Both of the above two points are suitable for our system. In details, detected faces are first aligned into canonical position, and then component-based local binary patterns [1] are extracted from the images to form feature vectors. After feature extraction, sparse representations are computed from these feature vectors using an L1-regularized least square objective function. Non-zero entries of sparse representations are considered as visual words for inverted indexing.

Due to the nature of faces, images of the same individual may have high intra-class variation. To leverage the partially-tagged identity information, a regularization term is added to the objective function to force images of the same identity (tag) to have similar sparse representations. These images will propagate visual words to each other, and the query image will be able to find all images of the same individual if it is similar to at least one of them.

By incorporating such framework into our system, in addition to attributes, the user can also use a face image itself as the face content.

[3]For example, the work of [16] has trained as many as 73 attribute classifiers.

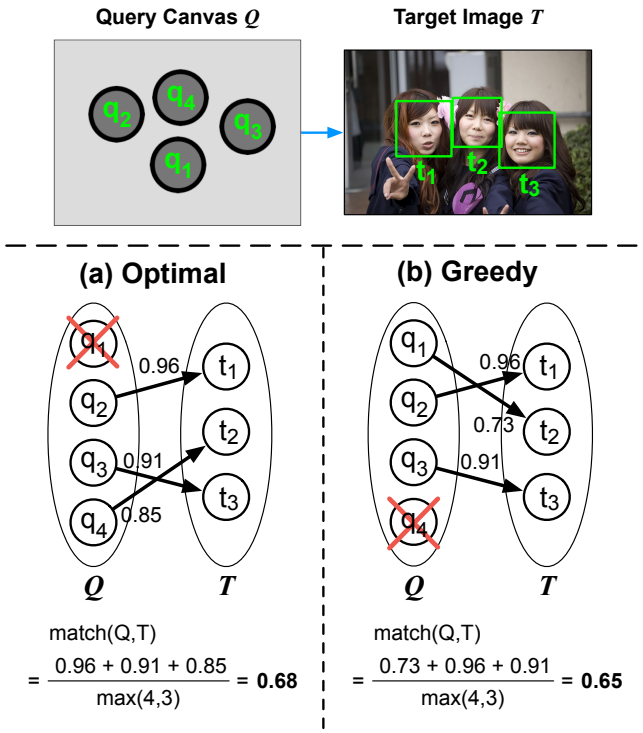

Figure 5: The image ranking problem as a maximum weighted bipartite matching between the query canvas (set Q) and the target image (set T). The numbering in the query canvas implies the order in which the faces are specified. The optimization in Eq. 1 can be carried out by (a) the optimal solution or (b) the greedy approximation. A red cross indicates a mismatched face, and $match(Q,T)$ means the overall matching score between Q and T.

5. IMAGE RETRIEVAL

5.1 Problem Formulation

5.1.1 *Maximum Weighted Bipartite Matchings*

As illustrated in Fig. 5, for a (query canvas, target image) pair, denoted as (Q,T), the image ranking problem is formu-

lated as a maximum weighted bipartite matching between the two sets Q and T. The objective function $match(Q,T)$, or the overall matching score between Q and T, is defined as the sum of the individual face matching scores $match(q,t)$ (defined in Sec. 5.2) divided by $\max(|Q|,|T|)$. The formulation is as the following constraint optimization problem:

$$match(Q,T) = \frac{\max\left[\sum_{q\in Q}\sum_{t\in T} match(q,t)\delta(q,t)\right]}{\max(|Q|,|T|)} \quad (1)$$

$$\text{subject to:} \quad \delta(q,t) \in \{0,1\} \qquad \forall q \in Q, t \in T \quad (2a)$$

$$\sum_{t\in T}\delta(q,t) \le 1 \qquad \forall q \in Q \quad (2b)$$

$$\sum_{q\in Q}\delta(q,t) \le 1 \qquad \forall t \in T \quad (2c)$$

$$match(q,t) > 0 \qquad \forall q \in Q, t \in T \quad (2d)$$

$$\delta(q,t) = \begin{cases} 1, & \text{if } (q,t) \text{ is matched} \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

where $\delta(q,t)$ (Eqs. 2a and 3) is an indicator variable of whether (q,t) is matched.

Note that the matching ensures each query face $q = q_1, ..., q_{|Q|}$ matches at most one target face $t = t_1, ..., t_{|T|}$ (Eq. 2b), and each t is matched at most once (Eq. 2c). We add the subscripts here to explicitly denote the individual faces in Q and T.

The numerator of Eq. 1 is the objective function in maximum weighted bipartite matchings. Note the $\max(|Q|,|T|)$ in the denominator. The positive weights (Eq. 2d) ensure that the number of matching pairs equals $\min(|Q|,|T|)$. If the numbers of faces in Q and T are the same, dividing by $|Q|$ or $|T|$ is like averaging. But if $|Q|$ and $|T|$ are different, the overall matching score will be divided by a larger number. Thus, this formulation much favors target images that have the same number of faces as the query canvas.

Fig. 5 shows an illustration of matching 4 query faces with 3 target faces. The optimal solution, by the above formulation, always comes up with the highest $match(Q,T)$ among all the possible matches. As in Fig. 5 (a), $match(Q,T) = 0.68$ for this example. However, computing the optimal solution (e.g., by the Bellman-Ford algorithm) is inefficient if we have to repeat for all target images.

5.1.2 Greedy Approximation

The inefficiency in solving Eq. 1 can be compromised by the proposed greedy approximation. By greedy, we mean the first query face q_1 is the first to match by choosing the best matching face remaining (i.e., the unmatched t^* for which $match(q_1,t^*)$ is maximized), followed by q_2, q_3, etc.. The numbering in Q implies the order in which the faces are specified on the query canvas. The procedure is summarized in Algorithm 1.

In the example of Fig. 5 (b), the greedy approximation allows the first query face q_1 to match first, choosing t_2 to get face matching score of 0.73. The second query face q_2 chooses t_1 of 0.96, followed by q_3 choosing the last target face t_3 remaining to get 0.91. q_4 then becomes a mismatched face (indicated by a red cross in Fig. 5 (b)), but it could have matched t_2 of 0.85 if specified in the first place. Eventually, $match(Q,T) = 0.65$ in the greedy approximation.

Algorithm 1 The procedure in greedy approximation.

Input: The query canvas Q and the target image T.
Output: The overall matching score $match(Q,T)$.

```
match(Q,T) ← 0
/* Maintain a remaining set R. */
R ← T
for q ← q_1, q_2, ..., q_|Q| do
  t* ← max_{t∈R} [match(q,t)]
  match(Q,T) ← match(Q,T) + match(q,t*)
  R ← R − {t*}
end for
match(Q,T) ← match(Q,T) / max(|Q|,|T|)
```

In general, although the greedy approximation has a lower $match(Q,T)$ than the optimal solution, it significantly reduces the computational cost and reflects the idea that the first face coming to the user's mind is the most important.

5.2 Face Matching Scores

Our work uses multimodal fusion to determine the face matching score $match(q,t)$ between a query face q and a target face t. It is defined as a linear combination of the matching scores for facial attributes, appearance similarity, face position, and face size:

$$\begin{aligned} match(q,t) = w_{attr}&\left(\prod_{\tau} Attr(q_\tau,t_\tau)\right)^{1/|\tau|} \\ &+ w_{app}App(q,t) \\ &+ w_{pos}Pos(q,t) + w_{size}Size(q,t) \end{aligned} \quad (4)$$

$$Attr(q_\tau,t_\tau) = \begin{cases} t_{\tau k}, & \text{if } q_\tau = k \\ 1.0, & \text{if } q_\tau = \text{not specified,} \end{cases} \quad (5)$$

where w_{attr}, w_{app}, w_{pos}, and w_{size} are the weights for these four modalities.

The first term in Eq. 4 weights the geometric mean of the matching scores $Attr(q_\tau,t_\tau)$ for all of the attribute types τ, i.e., gender, age, and race ($|\tau| = 3$). As in Eq. 5, if q_τ, or the attribute specification of the query face for type τ, is some attribute k, then $Attr(q_\tau,t_\tau) = t_{\tau k}$, the attribute score of k in type t. For instance, if q specifies the age "youth", then $Attr(q_{age},t_{age})$ takes the attribute score for youth of t. In notation, if for τ = age, q_{age} = youth, then $Attr(q_{age},t_{age}) = t_{age,youth}$.

In contrast, if for τ, the attribute is not specified, then $Attr(q_\tau,t_\tau) = 1.0$, the perfect score. The choice of geometric means rather than arithmetic means is to avoid outliers for some attribute type. The second term in Eq. 4 weights the appearance similarity score between q and t, obtained in Sec. 4.2. Note that in our user interface (Sec. 6.1), attributes and appearance similarity (by a specific face instance) of a query face are not specified at the same time.

5.3 Score Normalization

The real-valued scores of each of the four query modalities, that is, attributes, appearances, positions, and sizes, are normalized into the range $(0,1)$ for late fusion. 0 and 1 represent the worst score and the best score of a modality.

For an attribute score $t_{\tau k}$, we first normalize the strong classifier's output to zero mean and unit variance for each attribute k. Then we apply a sigmoid function to map it to

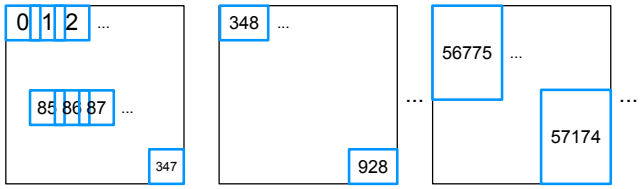

Figure 6: Quantization of (x', y', w', h') into overlapping blocks of various positions and sizes, where the four variables represent the already quantized horizontal and vertical positions, width, and height. The mapping between a (x', y', w', h') combination and a block ID should be unique throughout the system.

$(0, 1)$. The appearance similarity scores $App(q, t)$ are normalized in a similar way.

For the matching scores for face position $Pos(q, t)$ and face size $Size(q, t)$ between a query face q and a target face t, first note that in our system, coordinates are always represented as fractions of the width or height of the image (canvas). This fractional representation allows the computation to be adapted to the various aspect ratios in the the target images (query canvas). The definitions of $Pos(q, t)$ and $Size(q, t)$ are based on the distance errors between q and t as follows:

$$Pos(q, t) = 1 - \frac{d_{center}}{\sqrt{2}} \tag{6}$$

$$Size(q, t) = 1 - \frac{d_{width} + d_{height}}{2}, \tag{7}$$

where d_{center} is the L2 distance between the face centers, and d_{width} and d_{height} are the L1 differences between the face widths and heights. The denominators $\sqrt{2}$ and 2 in Eqs. 6 and 7 indicate the maximum (worst) distance between the face centers and the maximum width plus height differences between the faces, i.e., the diagonal line and the whole width plus whole height. Therefore, each term subtracted from 1 is now normalized into the range $(0, 1)$.

5.4 Block-based Indexing

We apply a block-based method to spatially index all the database faces. Since the face center coordinates, width and height, denoted as x, y, w, and h, are fractions, the infinitely many numbers in the interval $(0, 1)$ make indexing computationally infeasible and quantization too sensitive. Therefore, we first uniformly quantize each of the four variables into L levels, denoted as x', y', w', and h', each in the range $[0, L-1]$. We then quantize the valid (x', y', w', h') combinations uniquely into overlapping blocks of various positions and sizes, as illustrated in Fig. 6. Note that not all the L^4 combinations are valid (within-boundary) blocks. The mapping between an (x', y', w', h') tuple and a block ID should be unique throughout the system. One such mapping is easily achieved by representing the block ID as an L-nary number of 4 digits. For example:

$$BlockID = x' + y'L + w'L^2 + h'L^3. \tag{8}$$

The ordering of digits does not matter as long as it is consistent. The mapping[4] in Eq. 8 is not only unique but also reversible and storage-free (no table lookup).

[4]As an example, suppose $L = 20$ levels, each being 0.05. An (x, y, w, h) combination of (0.11, 0.28, 0.42, 0.67) will be quantized into $(x', y', w', h') = (2, 5, 8, 13)$. The block ID is then $2 + 5 \cdot 20 + 8 \cdot 20^2 + 13 \cdot 20^3 = 107302$. The reverse mapping can restore (x', y', w', h') from the block ID.

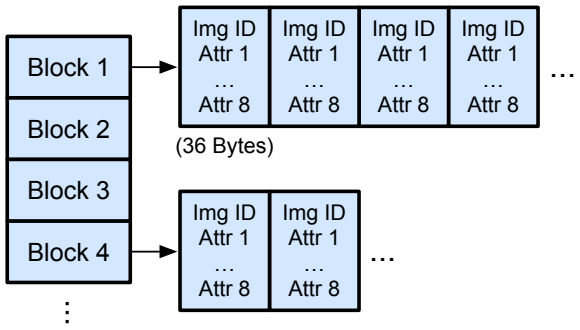

Figure 7: The indexing structure in the proposed system. The block IDs are treated as the visual words in typical inverted indexing. Each of them corresponds to an inverted list of structures, each being a tuple of (image ID, the 8 attribute scores) that requires 36 bytes in the implementation.

To build the index, the block IDs are treated as visual words in typical inverted indexing. Each block then corresponds to an inverted list of structures, each being a tuple of (image ID, the 8 attribute scores), as in Fig. 7. In other words, each list contains all the faces and their attribute scores within this particular block.

Since retrieving only faces in the block of the query face is still too sensitive, in the online search, a query face runs a "sliding window" to retrieve faces in W neighboring blocks. These neighbors are found by adjusting each of the (x', y', w', h') up and down for various quantization levels to produce new combinations. An example neighbor may be $(x' - 2, y' + 1, w' + 3, h')$. Then, we apply the mapping in Eq. 8 to get the neighboring block IDs and retrieve the corresponding inverted lists. The range of the sliding window, denoted by parameters tol_{pos} and tol_{size}, controls the level of tolerance in positions and sizes.

For multiple-face queries, each query face is processed separately to collect relevance scores from inverted lists according to Eq. 4. The greedy manner still applies that the first query face scans the inverted lists first. Finally, the results are merged into a ranking list according to Eq. 1. The retrieval results of block-based indexing and linear scan differ mostly by the quantization errors and the faces skipped by the sliding windows.

6. EXPERIMENTS

In this section, we describe the touch-based user interface of the proposed system named "Where is Who" (short for WiW), followed by the dataset and implementations. We also conduct an estimation on storage cost. For a video demonstration of the system, please visit our project page: `http://www.csie.ntu.edu.tw/~winston/projects/face/`

6.1 User Interface

The user interface of our system is shown in Figure 8. The user can drag faces from the top-right area onto the canvas

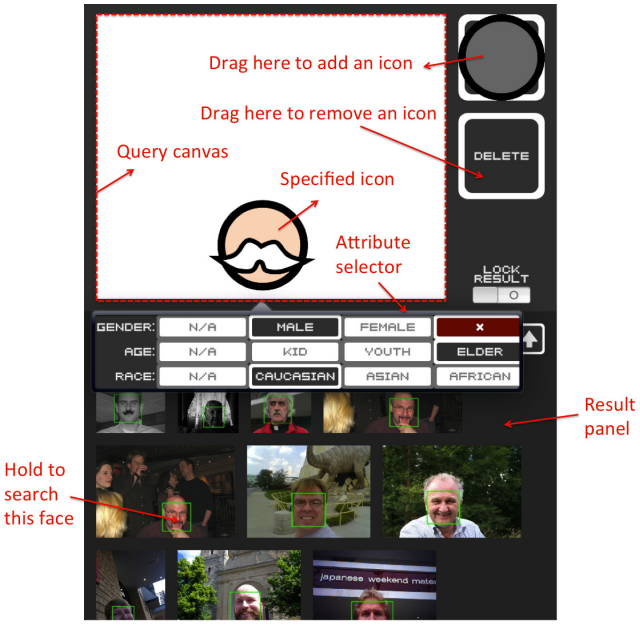

Figure 8: The touch-based interface of our system. The user can formulate a query by dragging face icons from the top-right area onto the canvas at desired positions. They can also pinch their fingers to adjust the sizes of the icons and the canvas. When holding an icon, a pop-up menu will show up for attribute selection. When browsing the search results on the bottom, they can also hold a face and use the changed icon on the top-right to find similar faces (appearance-based) in other photos. For every canvas modification, the system performs a search so that the user can refine their search intention interactively.

at desired positions. They can also pinch their fingers to adjust the sizes of the icons and the canvas. Holding an icon invokes a popup attribute selector. We have designed a total of 48 face icons ($3 \times 4 \times 4$) to represent the various attribute combinations. For appearance-based retrieval, the user can hold a face in the result panel and use the changed icon on the top-right to find similar faces in other photos. For every canvas modification, the system performs a search and shows the results on the bottom so that the user can refine their search intention interactively viewing the current results. Since our system is naturally suitable for a touch-based interface, we have implemented the UI on a tablet device.

6.2 Dataset and Implementations

The dataset is composed of two portions. As mentioned in Sec. 4.1, we crawl a large number of user-contributed photos from Flickr as the first (main) portion. For appearance-based retrieval, 732 daily photos containing 1,248 faces are added to the dataset as the second portion. After face detection by a public API [7], together there are $N = 115,487$ images in the dataset where the average number of faces per image is $F = 2.117$, so the dataset contains $N \times F = 244,491$ faces.

Since appearance-based retrieval is intended as a secondary feature of our system, we only estimate the appearance similarity scores in the second portion. Therefore, faces in the first portion always have zero appearance similarity scores if they are specified on the canvas.

In attribute detection, we adopt the LIBSVM software package [4] for learning the mid-level features. For the fusion weights in Eq. 4, we conduct a sensitivity test to select w_{attr}, w_{pos}, and w_{size} (that sum to 1) to optimize the evaluation criterion in Sec. 7.2. For block-based indexing, we empirically select the number of quantization levels as $L = 20$, and the tolerance (range) of the sliding window as $tol_{pos} = \pm 4$ levels and $tol_{size} = \pm 4$ levels[5].

The server part of WiW is implemented on a 16-core, 2.40GHz Intel Xeon machine with 48GB of RAM.

6.3 Storage Estimation

Since appearance-based retrieval is considered as a secondary feature, the storage cost of codeword index is not considered in this estimation. Following the format of the index structure in Fig. 7, for an inverted list structure, we require 4 bytes for an image ID and $4 \times 8 = 32$ bytes for the eight floating-point attribute scores. That is, 36 bytes for indexing a face. The cost of headers (block IDs and counts) can be neglected in the calculation. Multiplied by $N \times F$, it requires approximately $244.5K \times 36B = 8.8$MB in optimal implementations. Reusing $F = 2.117$ in our dataset, an 1-million image dataset requires a storage cost of around $1M \times 2.117 \times 36B = 76.2$MB.

7. PERFORMANCE EVALUATION

In this section, we evaluate the performance of several variants of our proposed system. We have conducted an experiment to evaluate known-item search, in which the user tries to search for a specific target image in mind. Since appearance-based retrieval is treated as a secondary feature, refer to [5] for the corresponding evaluation. We also evaluate the efficiency of indexing by measuring the running time and the number of visited faces.

7.1 Compared Methods

To the best of our knowledge, our system is the first work to address the problem of face image retrieval based on both facial attributes and face layout. So we compare four variants of the proposed system: (1) "Attr," by enabling only w_{attr} in Eq. 4 with linear scan in order to resemble the search scheme in [15], (2) "Pos + Size (index)," by enabling w_{pos} and w_{size} with block-based indexing (Sec. 5.4), (3) "Attr + Pos + Size," by enabling w_{attr}, w_{pos}, and w_{size} with linear scan, and (4) "Attr + Pos + Size (index)," same as (3) but with block-based indexing. (4) is the full version of WiW except for the appearance-based component.

[5]Therefore, a sliding window visits $W = (4 \cdot 2 + 1)^2 \cdot (4 \cdot 2 + 1)^2 = 6,561$ neighboring blocks. Many of the blocks may be out-of-boundary or empty.

Table 2: Distribution of the number of faces in the 500 query tasks.

# faces	1	2	3	4	5+	Total
# query tasks	249	147	55	22	27	500

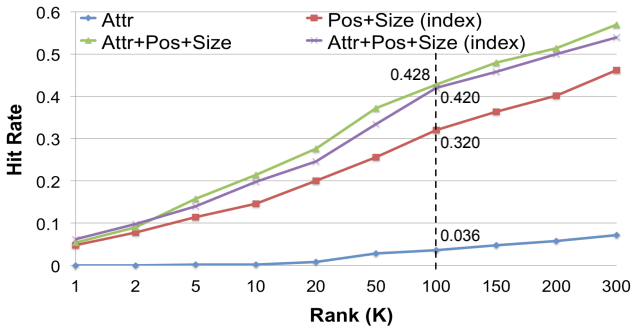

Figure 9: Hit rates@K of different methods over the 500 query tasks for known-item search. Adding layout information achieves a hit rate of 0.420 at rank 100 (purple line). This significantly outperforms 0.036 of using attributes alone (blue line), the search scheme proposed by [15]. In addition, adding attribute information improves the hit rate from 0.320 (red line) to 0.420 (purple line).

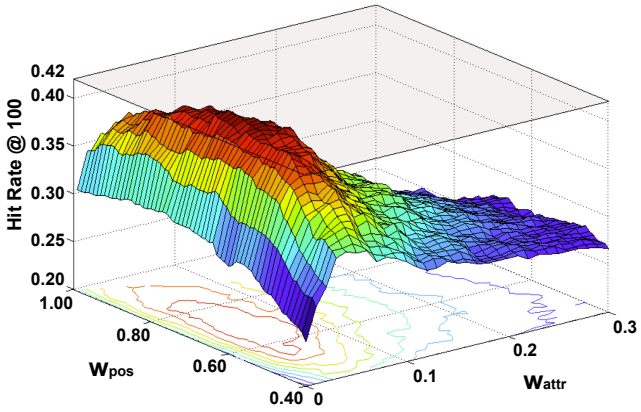

Figure 10: The fusion weight selection to maximize hit rate. The three axes represent w_{attr}, w_{pos} (in Eq. 4), and hit rate@100, respectively. The non-negative weights are constrained by $w_{attr} + w_{pos} + w_{size} = 1$.

7.2 Performance of Known-Item Search

7.2.1 Evaluation Setup

In known-item search (KIS), the user aims to search for a specific target image that they have seen. To simulate such a scenario in a large-scale dataset, 500 target images, each containing at least one face (985 faces in total), were randomly selected from our dataset (portions 1 and 2) as query tasks. The distribution of the number of faces is summarized in Table 2. These query tasks were equally distributed among the participants of 20 subjects invited to the experiment.

For each query task, the subject was asked to first carefully observe the target image, and then formulate a query canvas by graphically placing attributed icons at the corresponding positions and in the corresponding sizes for each query face. The subjects were asked to specify the positions and sizes according to the bounding boxes detected by the system in order to minimize the effect of face detection errors. The attributes were specified according to their "strengths" to the subject. If either the gender, age, or race of the face was not obvious enough, the attribute would be "not specified" for this type. Finally, the 500 submitted query canvases were collected for later evaluations.

Although this simulation does not reflect the reality that the user may not accurately remember the face layout or the face content in a large image collection over a long time, our user interface makes it easy to gradually refine the canvas by providing a real-time re-query for every canvas modification. This is useful in reality because the user usually performs several trials in the same way in typical retrieval systems.

7.2.2 Gain from Layout Information

To evaluate how well the target image is ranked in the results, we measure the "hit rate@K" as in [3][6], the proportion of the 500 query tasks where the system can retrieve the target image within the top K search results (within rank K).

Fig. 9 shows the performance of the four compared methods for all query tasks. Apparently, all three methods considering face layout significantly outperform "Attr," achieving hit rates of 0.320, 0.428, and 0.420 at rank 100[7] that are 8.8 to 11.8 times higher than 0.036 (blue line) of using attributes alone, the search scheme proposed by [15]. This clearly explains that when the user has the face layout in mind, specifying the positions and sizes on a canvas provides much more information than specifying only the face content.

Also, the hit rate of "Attr + Pos + Size (index)" (purple line) is slightly lower than its linear-scan variant "Attr + Pos + Size" (green line). This is due to the quantization errors introduced by the block-based indexing, where the exact positions and sizes are quantized into nearby blocks.

7.2.3 Gain from Attribute Information

We can also observe in Fig. 9 that "Attr + Pos + Size (index)" outperforms "Pos + Size (index)." In other words, adding attribute information can further improve the hit rate@100 from 0.320 (red line) to 0.420 (purple line), although in the fusion weight selection (Fig. 10), the contribution of w_{attr} is only 5% of the total weight. The small weight can be explained by the fact that attribute detection is less robust than face detection and localization.

As reported in Sec. 4.1, a single attribute detector has an average accuracy of around 80%, but when three attributes are specified in a query face, we can expect only $(0.80)^3 = 51\%$ of the target faces to have all correctly de-

[6]In KIS, the performance is often measured by mean reciprocal rank. However, because there may be numerous other images with similar face content and face layout, especially images of 1 or 2 faces (Table 2), many of our target images are ranked up to number several thousand. Averaging by those near-zero reciprocal ranks would make it difficult to compare different methods.

[7]Although a hit rate of 0.420 at rank 100 may not be high enough for practical photo management, the images returned by the system are often relevant to the query canvas, as illustrated in Fig. 13. This high precision enables casual photo browsing when the user does not have a specific target in mind.

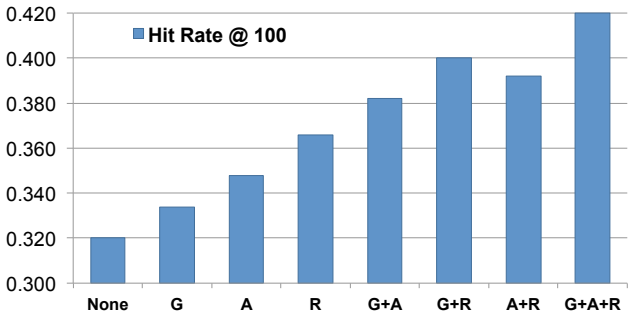

Figure 11: Breakdown of the improvement of hit rate@100 from 0.320 to 0.420 (Fig. 9) by enabling different combinations of attribute types. G, A, and R stand for gender, age, and race, respectively. Starting from 0.320 of no attributes, we can observe that enabling more attribute types improves the hit rate towards the highest 0.420 (G+A+R).

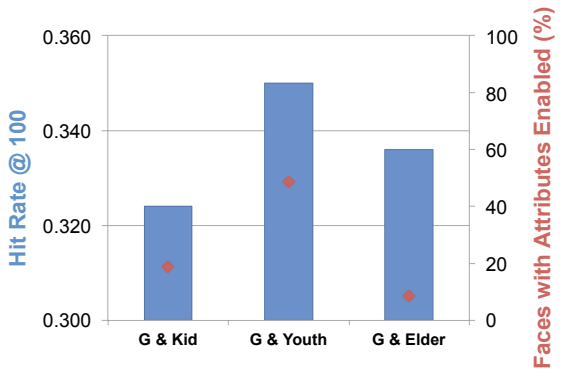

Figure 12: Hit rates@100 by *simultaneously* enabling any gender attribute and one age attribute. A red dot represents the percentage of query faces with such attributes enabled. We can observe that "G & kid" performs the worst among all alternatives, even worse than "G & elder" that has fewer faces with attributes enabled. This reflects the intuition that it is relatively hard to tell the gender among kids.

tected attributes. This is challenging for a system supporting multi-attribute queries. Also, in the experiment of KIS, the subjects were instructed to specify the positions and sizes according to the bounding boxes returned by the face detector. This accurate layout information has compromised the contribution of attributes in the multimodal fusion.

7.2.4 *Breakdown by Attribute Combinations*

Following Sec. 7.2.3, we break down the improvement of hit rate@100 from 0.320 to 0.420 (Fig. 9) by enabling different attribute combinations. In Fig. 11, G, A, and R stand for the three attribute types gender, age, and race, respectively. "A + R," for example, enables attributes in faces in the query tasks where the user specified any age attribute *or* any race attribute.

Starting from 0.320 of no attributes, we can observe that enabling more attribute types improves the hit rate towards the highest 0.420 (G + A + R). Again, this shows the power of multi-modality in retrieval systems. With more attributes available, such as the 73 detected attributes in [16], we can expect such a system to achieve even better performance for practical usage.

It is also interesting to discuss the effect of using some attributes together in a query canvas. Fig. 12 shows the hit rates@100 by *simultaneously* enabling (hence the "&" symbol) any gender attribute (G) *and* one age attribute. A red dot counts the percentage of faces with attributes that meet this requirement.

Generally, the higher the red dot, the more faces with attributes enabled, which is expected to raise the hit rate. However, we can observe that "G & kid" performs the worst among all alternatives, even worse than "G & elder" that has fewer faces with attributes. It reflects the intuition that it is relatively hard to tell the gender among kids.

7.3 Efficiency of Indexing

From the 500 query tasks, we also record the average running time and the average number of visited faces, including repetitive visits, in the search. Table 3 shows the efficiency comparison between linear scan and block-based indexing. In both manners, block-based indexing speeds up around 3 times and requires only 0.0558 second in a dataset of more than 200k faces. Although there is still room for improvement (e.g., incorporating attribute scores into the visual words, or better quantization and search methods for the (x,y,w,h) information), we believe the proposed indexing method can be extended to a million-scale dataset.

Table 3: The efficiency comparison between linear scan and block-based indexing, measured by the average running time and the average number of visited faces (including repetitive visits) in the search.

	Running time (sec)	# Visited faces
Linear scan	0.2089	331,225
Block-based index	0.0558	111,303
Indexing speedup	3.74x	2.98x

8. CONCLUSIONS AND FUTURE WORK

Our work proposes a novel way to effectively organize and search for consumer photos by placing attributed face icons on a query canvas at desired positions and in desired sizes. With the help of automatic facial attribute detection and appearance similarity estimation in the offline process, we are able to analyze wild photos without tagging at all. In the online process, the system simultaneously considers attributes, appearances, positions, and sizes of the target faces.

The scenario has been realized on a tablet device with a touch interface. Experimenting with a large-scale Flickr dataset of more 200k faces, we have achieved a hit rate@100 of 0.420, significantly improving from 0.036 of prior search scheme [15] using attributes alone. We have also achieved a fast retrieval response of 0.0558 second by the proposed block-based indexing approach. Experimental results from extensive search tasks (Fig. 13) reveal the potential for effective and efficient photo management.

In the future work, we will exploit more facial attributes for the proposed search system. We will also include more

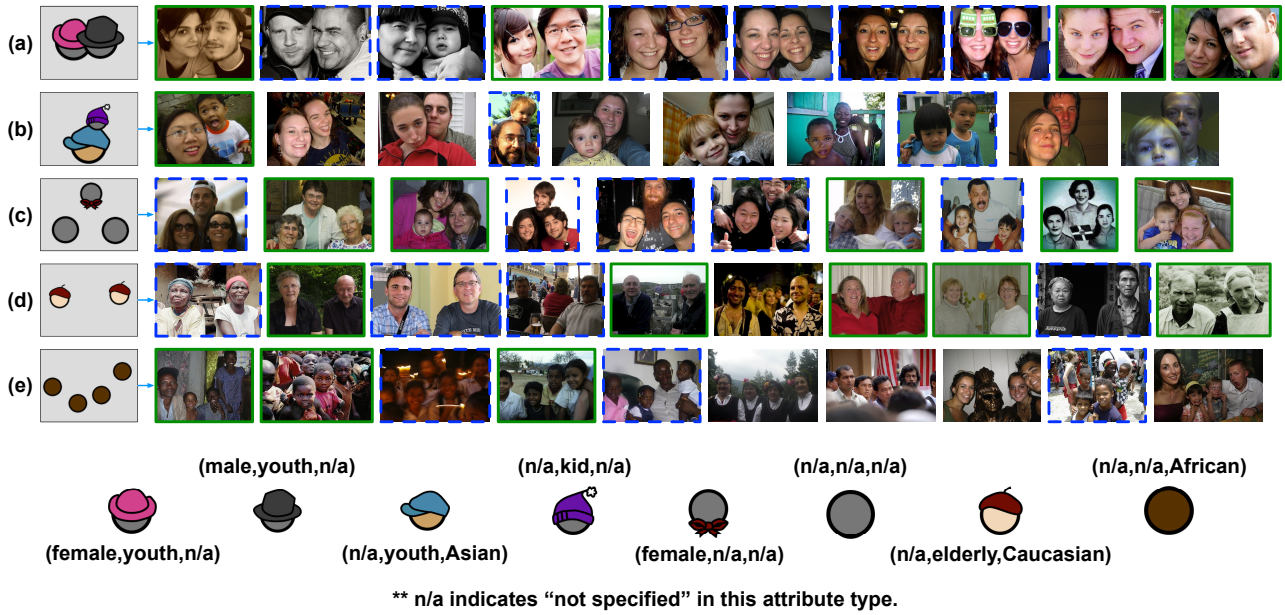

Figure 13: Example query canvases and the corresponding top 10 search results. The figure demonstrates extensive search tasks ranging from very close faces ((a) and (b)) to faces spread in various ways ((c), (d), and (e)). The icons representing the attribute combinations are shown in the bottom. Images squared in green solid lines (blue dashed lines) are believed to be highly (partially) relevant by an average user.

context cues (e.g., time, geo-locations, etc.) for consumer photo management. Meanwhile, the human factors will be considered more in the integration with mobile devices.

9. REFERENCES

[1] T. Ahonen, A. Hadid, and M. Pietikainen. Face description with local binary patterns: Application to face recognition. *PAMI*, 2006.

[2] M. Ames and M. Naaman. Why we tag: Motivations for annotation in mobile and online media. *ACM CHI*, 2007.

[3] Y. Cao, C. Wang, L. Zhang, and L. Zhang. Edgel index for large-scale sketch-based image search. *CVPR*, 2011.

[4] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[5] B.-C. Chen, Y.-H. Kuo, Y.-Y. Chen, K.-Y. Chu, and W. Hsu. Semi-supervised face image retrieval using sparse coding with identity constraint. *ACM Multimedia*, 2011.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005.

[7] face.com API. `http://developers.face.com/`.

[8] M. Freeman. *The Photographer's Eye: Composition and Design for Better Digital Photos*. Focal Press, 2007.

[9] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory*, 1995.

[10] iPhoto from Apple Inc. `http://www.apple.com/ilife/iphoto/`.

[11] P. Isola, J. Xiao, A. Torralba, and A. Oliva. What makes an image memorable? *CVPR*, 2011.

[12] L. Kennedy, M. Naaman, S. Ahern, R. Nair, and T. Rattenbury. How flickr helps us make sense of the world: Context and content in community-contributed media collections. *ACM Multimedia*, 2007.

[13] L. S. Kennedy, S.-F. Chang, and I. V. Kozintsev. To search or to label?: Predicting the performance of search-based automatic image classifiers. *ACM MIR Workshop*, 2006.

[14] H.-N. Kim, A. E. Saddik, K.-S. Lee, Y.-H. Lee, and G.-S. Jo. Photo search in a personal photo diary by drawing face position with people tagging. *IUI*, 2011.

[15] N. Kumar, P. N. Belhumeur, and S. K. Nayar. Facetracer: A search engine for large collections of images with faces. *ECCV*, 2008.

[16] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Describable visual attributes for face verification and image search. *PAMI*, 2011.

[17] K.-S. Lee, J.-G. Jung, K.-J. Oh, and G.-S. Jo. U2mind: Visual semantic relationships query for retrieving photos in social network. *ACIIDS*, 2011.

[18] Picasa from Google Inc. `http://picasa.google.com`.

[19] H. Xu, J. Wang, X.-S. Hua, and S. Li. Image search by concept map. *SIGIR*, 2010.