Discovering Relevant Concepts for Web Image Search

Ruey-Cheng Chen[1] and Pu-Jen Cheng[2]

[1]*Institute of Information Science, Academia Sinica, Taiwan*

[2]*Dept. of Computer Science and Information Engineering, National Taiwan University, Taiwan*

[1]*cobain@iis.sinica.edu.tw*, [2]*pjcheng@csie.ntu.edu.tw*

ABSTRACT

It is difficult for users to browse the images retrieved from conventional Web search engines. The paper presents an approach that organizes the retrieved images with auto-generated relevant concepts to users' queries. The proposed approach is a well-integrated set of techniques, including relevant concept finding, query clustering, and query classification. The experimental results show that the performance of Web image retrieval can be effectively improved with the proposed approach.

1: INTRODUCTION

With the ease of capturing and publishing digital images, the amount of Web images has increased tremendously. One popular way to access Web images is to use search engines [5]. Most image search engines display a list of thumbnails of images matching users' search criteria; however, the returned images are not well organized. There is often a lack of conceptually meaningful presentation. Users may find it difficult to understand the structure of the concepts related to the query in the search results.

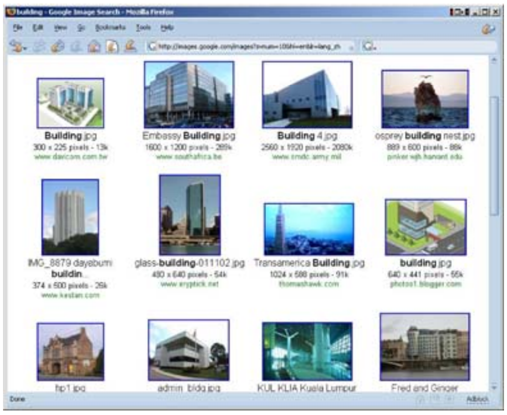

Figure 1: Images retrieved from Google Images.

For example, Figure 1 shows the images retrieved from Google Images (http://images.google.com) in response to a query about "buildings". The images are simply listed. Users have to go through the list to identify the results. When a given query has multiple sub-topics, it is believed that users cannot easily explore the concepts mixed together within the retrieved images.

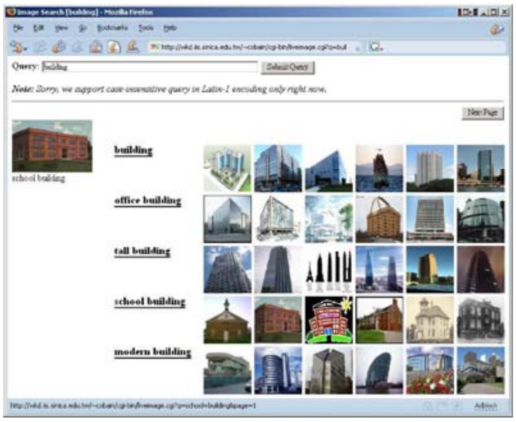

Figure 2: Images retrieved for "building" with our approach.

Figure 3: Images retrieved by clicking an image in the category of "office building" in Figure 2.

To assist users, the paper presents a new approach for them to find the images they want in a conceptually simple way that organizes retrieved images on the basis of auto-generated relevant concepts. With the generated relevant concepts, users have a comprehensive overview about the returned images and are able to locate the images of interest easily. Figure 2 shows the idea of the proposed approach. We generate relevant concepts such as "office building", "tall building", "school building" and "modern building" for the given query "building". The images listed in the retrieved results are organized according to their similarity to the relevant concepts, which makes it easier for users to locate images of interest with the corresponding concept names and relevant images. Users can then find other images with relevant concepts by simply clicking the images they prefer. To illustrate, Figure 3 shows the images further retrieved by clicking the category of "office building". Some relevant concepts such as "small office building", "federal office building", "office building clips", and "cartoon office building" are generated for the clicked images with the associated relevant images.

The purpose of this paper is to discover meaningful concepts relevant to image queries from search engine logs. We are interested in developing an automatic approach that organizes users' query terms into auto–generated subject categories. As users' queries are short and new queries appear all the time, our problem is to assign effective concept categories to users' queries and improve their search performance, even if the given queries have never appeared before.

The experiment results show that the performance of Web image retrieval can be effectively improved with the proposed approach.

The rest of the paper is organized as follows. The related works are shown in Section 2. In Section 3, we elaborate the proposed approach for searching images by relevant concepts. Then we conducted several experiments for analyzing our approaches and system benefits in Section 4. The conclusions are drawn in Section 5.

2: RELATED WORK

Search results clustering for improving Web search has been investigated in numerous works recently. Most of them were focused on clustering search results for document searching [7,13,14]. They aimed to extract clusters from search-result snippets returned by text search engines. Although clustering search-result snippets is fast enough for online searching, the generated clusters are inadequate for browsing images as the ways to annotate images are often quite different from those to describe documents. Some works extended the technology to image retrieval. For example, [3] analyzed the co-occurrence between terms associated with image captions and used a statistical relation, called subsumption, to hierarchically organize the terms into clusters. However, many terms in the captions, e.g. person and location names, can not well organized into a taxonomy.

Lots of investigations have been explored to organize images into topic classes based on textural and/or visual information about Web images. Some works applied traditional content-based image retrieval (CBIR) techniques to cluster retrieved images. [9] located images to a given query-image using a color feature and clustered the top-ranked images with a hierarchical agglomerative clustering method. The clusters were then ranked according to their similarity to the query-image. [6] proposed an automatic method for hierarchical classification of images via supervised learning. The CBIR-based methods face the scalability problem. Moreover, semantically-relevant images may not have similar visual features. Recent research results [11] revealed the arrangement of images by semantic similarity is more useful than by visual similarity.

Some works took into account both of the textual and visual information. [2] partitioned a Web document into several blocks and then extracted textual and link information of an image from the block containing the image. The similarity of two images was determined by the combination of textual, visual, and block-level link features. [8] developed a image search engine, called AMORE, that grouped retrieved images by their URLs, which may give an indication of the types of the images it contains, their surrounding texts and their low-level image features such as color and object layout. [1] presented a statistical model for hierarchically modeling the statistics of word and feature occurrence and co-occurrence and for organizing image collections which simultaneously integrated semantic and visual information. This model supported not only automatic image classification but association of terms with images. [12] performed a semi-automated classification of Web images into a topic hierarchy based on associated texts and URLs. These works exploit various characteristics of Web images to cluster visually-similar or semantically-relevant images. This paper focuses mainly on grouping retrieved images using users' queries obtained from search engine logs.

3: DISCOVERY OF RELEVANT CONCEPTS

3.1: OVERVIEW

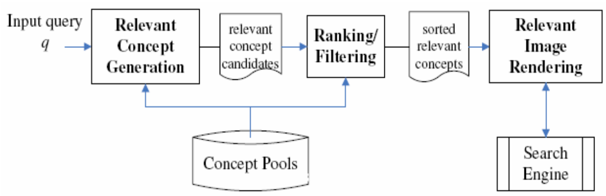

Figure 4: Overview of the proposed approach.

We propose an approach that generates a set of concepts relevant to a given query and returns images related to the concepts. Given a query q, our goal to obtain a set of relevant concepts $R(q)$ can be summarized as Figure 4. First, in the *Relevant Concept Generation* stage, a large numbers of concepts are found/derived from a concept pool and are treated as relevant concept candidates, where the *concept pool* is built from query logs and can be seen as a knowledge base. Next, the *Ranking/Filtering* process is applied so that each relevant concept candidate is assigned a score, called *interestingness*, to estimate the degree of users' interest and to prune the candidates which seem to irrelevant to the query q. Finally, after the Ranking/Filtering stage, we have the sorted relevant concepts, $R(q)$, of a given query q so that we can proceed the *Relevant Image Rendering* process that performs image search q by showing its relevant images with the concepts in $R(q)$.

3.2: RELEVANT CONCEPT

GENERATION

In this stage, we are going to obtain a set of *relevant concept candidates R'(q)* from a concept pool, given an input query *q*. The concept pool is a set of concepts, denoted as $P=\{c_1, c_2, \ldots, c_{|P|}\}$, containing a large numbers of concepts extracted mainly from query logs. The relevant concept candidates $R'(q)$ can be obtained with two different strategies: by elaboration and by inheritance, respectively. The result is then obtained by union of both strategies.

Generating by Elaboration

Given a query *q*, we generate relevant concept candidates by looking up *P* to see if there are concepts elaborating *q*, that is,

$$\{x|q \rightsquigarrow x, \forall x \in P\},$$

where $q \rightsquigarrow x$ means *x* elaborates *q*.

The elaboration relation can be either implicit in semantic level, for example "building" $\rightsquigarrow$ "skyscraper"; or explicit in term composition level, like "building" $\rightsquigarrow$ "office building". In this paper, we use only the latter method for simplicity reason. That is, $q \rightsquigarrow x$ holds iff *q* finds a match in *x*.

Generating by Inheritance

If we have all possible image queries of the world, we can just use the elaboration method mentioned above to generate relevant concept candidates. However, this is not always the truth. Due to the insufficient coverage of a query log, we propose an approach to compensate the sparseness nature of it.

The core concept of the approach is: "Conceptually similar queries have similar usages of modifiers". The term modifier means a word (or words) used to modify or qualify a target word. For instance, let "building" be a target word, "tall" and "modern" are modifiers of "building".

Before going deeper, an example is taken to give a clear overview of the approach. Suppose the set of candidates generated by elaboration for "building" is {"office building", "tall building", "school building", "modern building", …} from the query log. Now we have a new query *q* ="skyscraper", which is unseen in the log. First, we recognize that *q* is relevant to the concept "building" by estimating their similarity; then we rewrite the set of candidates with {"office skyscraper", "tall skyscraper", "school skyscraper", "modern skyscraper", …}.

To perform the generation by inheritance, we need to find out conceptually relevant terms. The intuitive way to recognize if a query *q* is relevant to a concept is to calculate a distance matrix between the given query and all terms in concept pool *P*. The procedure sounds for the first time, but it is too time consuming if *P* is large. So in the very first step, given a query set, we gather conceptually similar terms (concepts) into groups. The group has terms with similar concept are called concept-class, denoted as *C*. The procedure that constructs concept-class is done by query clustering. The reason why we cluster queries is that we want to obtain suitable relevant concepts *R*(*q*) given an unseen query *q* by borrowing modifiers. The next step we should deal with is to find most suitable cluster *C* for each query *q*, which is called query classification. The construction of concept-class can reduce the number of calculation. We will explain query clustering and classification later in Section 3.3.

After we classify *q* to concept-class C_q, named *Relevant Concept-Class* of query *q*, we analyze the relevant concepts of C_q. We then extract the patterns that generating the relevant concept-class and rewrite the patterns so that we finally obtain the relevant concept candidates *R'*(*q*) for query *q*. To illustrate, after we classify q ="skyscraper" into relevant concept-class C_q ={"building", "house",…}, we have to find out patterns that elaborate each class *c* in C_q such as "tall *", "modern *", and "office *". After that, we rewrite patterns using most intuitive way by simply substituting "*" with *q*. Finally, we have the relevant concept candidates *R'*("skyscraper") = {"tall skyscraper", "modern skyscraper", "office skyscraper", …}.

We name the overall task of the step-by-step procedures illustrated above as "Relevant Concept Inheritance" because the relevant concept is inherited from the concept-class a query belongs to.

3.3: QUERY CLUSTERING AND CLASSIFICATION

To construct concept-class, we adopt a query clustering method proposed in [4], which extends a hierarchical agglomerative clustering algorithm to cluster query terms based on the vector-space model in information retrieval. In order to judge the similarity between two query terms, the method uses the Web as an extra training corpus. The basic idea is to estimate the similarity between the search-result snippets of the two terms. In other words, each query term is submitted to a search engine. Character/word bi- and tri-grams are then used together to extract feature terms from the retrieved snippets and top-*N* most-frequent feature terms are chosen to be the term vocabulary. With TF-IDF weighting scheme, each query term will be represented as a feature vector. The similarity is estimated by a cosine measure.

To classify query *q*, we first try to represent it as a feature vector like the feature extraction method used in query clustering. Suppose we have clustering result $QC=\{C_1, C_2, \ldots, C_{|QC|}\}$. We calculate the similarity between query *q* and all query terms in cluster C_i. After that, we get top-*k* most similar terms list $[st_1, st_2, \ldots, st_k]$ in descending order. Finally, query *q* is classified to cluster *C* by

$$C(q) = \arg\max_{\forall C_i} \sum_{j=1}^{k} \delta(C(st_j), C_i) w_{rank}(j),$$

where $\delta(i, j)$ is Kronecker delta: $\delta(i, j) = 1$ if $i = j$, otherwise, $\delta(i, j) = 0$; and wrank(j) is a score for each rank j; here we use linear descended method: wrank(j) = k-j +1 if $1 \le j \le k$, otherwise, wrank(j) = 0.

3.4: RELEVANT CONCEPT RANKING AND FILTERING

There are a few more processing steps after we generate the candidate set. First, we rank the candidates based on their degree of interest to users. As a result of applying the inheritance technique in the previous stage, it is inevitable that we may induce noise as well; therefore, we need a filtering mechanism to remove the noise. We now explain the two tasks in detail.

The major challenge of the ranking task is how to quantify the interest of a candidate. We assume that the popularity, namely, the number of occurrences of a candidate in the concept pool, reflects users' interests. This is reasonable, since our concept pool is constructed primarily from query logs. However, for an unseen candidate generated by inheritance only, its popularity is practically zero in the pool. To overcome the problem, we propose a measure called potential popularity. Consider the case where a candidate is generated by pattern. According to our procedure, there are a number of concepts in the same concept class that can derive the pattern; such concepts are called "voters". For example, suppose "cute dog" and "cute cat" are in the same concept class, then both are voters of the pattern "cute *". Since a candidate is derived from voters, it is reasonable to consider these voters and the similarity measure between voters x and candidate r when establishing an estimation of the interest of a candidate.

Our model is quite straightforward. Given an input query q and a candidate r, the voters involved in the generation of r are defined by

$$V(q,r) = \{y | x \in Class(q), y \in \kappa(x), \rho_{x \to q}(\{y\}) = r\}.$$

Each individual voter influences r according to its own popularity, while the influence is proportion to the similarity measure between r and x. Thus, the potential popularity of r can be formulated as

$$\sum_{\forall x \in V(q,r)} f_x Sim(x,r)$$

where f_x is the popularity of x; and $Sim(x,r)$ is the cosine similarity between x and r.

Combining r's self popularity measure with the potential popularity contributed by its voters, the interest score of a candidate r with an input query q is calculated by

$$I(q,r) = f_r + \sum_{\forall x \in V(q,r)} f_x Sim(x,r).$$

The next problem we need to deal with is the removal of noise from the candidate set. Some heuristic rules can help us alleviate the problem. Presumably noise is of less interest to users and can be safely ignored by setting up a predefined threshold. However, although removing candidates with lower content-coverage is a good idea, it is costly in practice and risk the performance by deleting concepts, such as person names. To prevent this possibility, filtering out noise is performed in two steps: First we remove candidate r if $Sim(x,r) < \theta_S$, where θ_S is the predefined threshold. Then, we remove candidates with $I(q,r) < \theta_I$, where θ_I is the predefined threshold.

3.5: RENDERING

The task can be divided into three steps: 1) sending a concept to the search engine; 2) extracting the value of the field "Number of Images" in the result page; and 3) extracting the URLs of images in the result page. Note that concepts with zero images returned are regarded as noise and are removed immediately. In practice, the rendering task is implemented as a standalone process to minimize the overhead of bringing up and down a network connection with the search engine.

4: EXPERIMENTS

In this section, we did an evaluation on query classification and then conducted a user study to evaluate the performance of the approach under a real-world scenario in which users attempt to find a set of images without predetermined targets. The focus of this survey is placed on the comparison between two different types of result presentation schemes – Google Images and the proposed approach – and how they guide users to find images of their own interests.

Data Set

Table 1: Query set (classified by types)

Animals & Plants	flower, dog, fruit
Computer-related	desktop, icon, background
Entertainment	Harisu, Brad Pitt, Ryoko Hirosue
Animation & Comics	game, Dragon Ball, Mashimaro
People-related	love, doctor, students
Nature	moon, autumn
Sports	basketball, World Cup
Shopping	Purin, cake
Gerography & Travel	Shanghai, zoo
Society & Culture	Valentine's Day, home
Transportation	train, Benz
Science& Tech.	mathematics

All unique queries are Capitalized.

In the experiments, we used the image-query log owned by VisionNEXT (http://www.visionnext.com) from June to September 2002, which had about 2 millions of queries (0.3 million distinct ones). To observe the performance of different types of queries, we created a balanced query set as follows: 100 queries were randomly selected from the most frequent 1,000 queries of the log, of which 30 were carefully selected as balanced query set by human assessors. The query set was built in accordance with the following criteria: 1) it covers thirteen query types proposed in [10]; 2) it contains queries appearing in the concept-class (seen) and queries not in the concept-class (unseen) in the ratio of 2 : 1. Our query set with each query's type is listed in Table 1.

Ten volunteers, who were experienced in text and image search, participated in our evaluation. They were requested to give a 5-grade score. Score 1 meant no relevance between the query and its given concept class, while score 5 meant highly relevant. It should be noted that the standard deviations of the results of the 10 subjects were between 0 and 0.86, with most below 0.5. This indicates that the subjects of this experiment agreed with each other.

Query Classification

Figure 5 shows the average scores (degree of relevance) for every subject and every query. The queries are grouped according to seen(Q_s)/unseen(Q_u) groups and sorted in ascending order by their degree of relevance. Comparing the overall performance between Q_s and Q_u, we observe that Q_s performs better. The performance of an unseen query is degraded if 1) there is no suitable concept for the query; or 2) subjects might give queries Q_s a higher grade (because those queries trivially exist in their relevant concept-classes, which implicitly affects the experiment subjects' judgment) so that queries Q_u have relatively low grades.

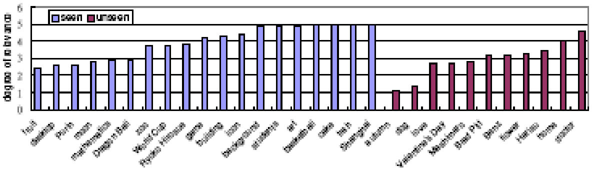

Figure 5: Performance of query classification.

Image Search

Table 2: Query-focused questions

Q1	Is the presentation scheme helpful in understanding the concepts related to the query?
Q2	Do images provide enough information to help you refine the query?
Q3	Do text descriptions of images provide enough information to help you refine the query?

Table 3: Performance-related questions

G1	Are you satisfied with the response time?
G2	Is the search interface suitable for browsing Web images?
G3	Is the search interface suitable for locating specific Web images?
G4	Is the search interface easy to use?
G5	Is the retrieved images organized in a semantically meaningful way?
G6	Is the retrieved images organized in a visually comfortable way?
G7	Briefly comment on the search interface.

Table 4: The results of query-focused questions

	Q1	Q2	Q3
GI-NE	2.77	2.43	2.87
LI-NE	3.69	2.78	3.65
GI-Common	2.85	2.74	2.91
LI-Common	4.11	3.58	4.01
GI-All	2.83	2.63	2.89
LI-All	3.97	3.31	3.89

Table 5: The results of performance-related questions

	G1	G2	G3	G4	G5	G6
GI	4.7	1.96	2.91	4.3	2.48	3.57
LI	3.35	4.09	2.35	3.96	4.04	3.48

In this experiment, all 10 subjects were asked to finish 30 predefined plus 10 free queries. In each query session, subjects were requested to search for Web images of their interest by using a prototype system, called LiveImage (LI) developed based on our approach, and Google Images (GI); then, they needed to answer 3 query-focused questions, as shown in Table 2, for each system after finishing the search. When they completed all the 30 query sessions, they gave scores for each system in several performance-related aspects, as shown in Table 3.

We list the average scores for each query-focused question in Table 4, where each row represents a combination of a system and a group of queries, i.e. named-entities (-NE,) common queries (-Common,) or all the queries (-All.) The results show that, in all three questions, the scores of GI-All are lower than those of LI-All; however, the difference between the systems in the question Q2 is not as great as that in the question Q1 and Q3. This is probably because 1) it is easier for the users to understand the structure of the search-results from a categorized view and thus easier to refine their queries, 2) it takes less effort for users to refine their queries according to the text descriptions rather than image content, and 3) the text descriptions of images returned by LI, primarily from query logs, contain less noise than the surrounding texts returned by GI.

In the first 4 rows, the performance of both systems on named-entities is lower than that on common queries. We also find that the difference between the scores of GI-NE and those of GI-Common is not obvious, while LI-Common outperforms LI-NE significantly. An interpretation for this phenomenon is that categorized search-results of a common query may greatly improve browsing experience because the images of a common query, such as “flower” or “cake”, contain a great number of diverse relevant concepts from which users can easily distinguish.

Finally, the average scores for general questions G1 to G6 are listed in Table 5. Concerning the response time, GI gets a score of 4.7, while LI gets only 3.35 in the question G1. Besides, GI is considered to be easier to use (4.3 in G4) and to have its search-results arranged in a visually-comfortable way (3.57 in G6.) On the contrary, LI is more suitable for browsing task than GI is (4.09 in G2) and organizes the search results based on their semantics (4.04 in G5). From the data, we can conclude that LI assists users indeed and the search results are more satisfying. When users are clear and confident of what they want from the search, the results from LI provide more related images and contain more information in both images and text descriptions than GI. Users can, therefore, choose a sub-topic in a shorter time. Furthermore, the presentation scheme of LI assists users even progressively while users have only vague concepts or uncertain ideas to what they want because LI offers users classified answers with both images and text descriptions. Users are given more information to find out and focus on the target images they want. On the contrary, we can see that the problem GI suffers in GI-Common is that once the users cannot search images in a specific term, GI can only order all the result images by how frequently they were searched before; however, this presentation scheme confuses and burdens users even worse. Therefore, LI that re-organize the result images improves the search interface.

5: CONCLUSION

According to [10], entertainment is the first motivation of image search interests; therefore, a retrieval system with impressive image search results will enjoy the users. The contemporary technology of image search, CBIR, is able to group images with visual similarity although it suffers from the limitations of retrieval accuracy and scalability. On the contrary, our approach organizes the image-search results into topic classes by mining relevant concepts to the queries of the users from search engine logs. The clustered topics not only assist users browse the retrieved images but also attract their notice by exposing interesting ones with respect to the queries.

In this paper, we have proposed an approach that organizes retrieved images with auto-generated relevant concepts. The experimental results demonstrate the feasibility and the effectiveness in improving Web image search.

REFERENCES

[1] K. Barnard and D. Forsyth. Learning the semantics of words and pictures. In *Proc. of the 8th International Conference on Computer Vision* 2, pp. 408-415, 2001.

[2] D. Cai, X. He, Z. Li, W.-Y. Ma, and J.-R. Wen. Hierarchical clustering of WWW image search results using visual, textual and link information. In *Proc. of ACM Multimedia*, pp. 952-959, 2004.

[3] P. Clough, H. Joho, and M. Sanderson. Automatically organizing images using concept hierarchies. In *Proc. of the SIGIR Workshop on Multimedia Information Retrieval*, 2005.

[4] P.-J. Cheng and L.-F. Chien. Auto-generation of topic hierarchies for Web images from users' perspectives. In *Proc. of the 12th ACM International Conference on Information and Knowledge Management*, pp. 544-547, 2003.

[4] C. Frankel, M. J. Swain, and V. Athitsos. WebSeer: An image search engine for the World Wide Web. *Technical report*, University of Chicago, Chicago, IL, USA, 1996.

[5] J. Huang, S. R. Kumar, and R. Zabih. An automatic hierarchical image classification scheme. In *Proc. of the sixth ACM international conference on Multimedia*, pp. 219-228, 1998.

[6] K. Kummamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *Proc. of the World Wide Web conference*, pp. 658-665, 2004.

[7] S. Mukherjea, K. Hirata, and Y. Hara. AMORE: a world-wide web image retrieval engine. In *CHI '99: CHI '99 extended abstracts on Human factors in computing systems*, pp. 17-18, 1999.

[8] G. Park, Y. Baek, and H.-K. Lee. Re-ranking algorithm using post-retrieval clustering for content-based image retrieval. *Information Processing and Management* 41(2), pp. 177-194, 2005.

[9] H.-T. Pu. A comparative analysis of web image and textual queries. *Online Information Review*, 29(5), pp. 457-467, 2005.

[10] K. Rodden, W. Basalaj, D. Sinclair, and K. Wood. Does organisation by similarity assist image browsing? In *CHI '01: Proc. of the SIGCHI conference on Human factors in computing systems*, pp. 190-197, 2001.

[11] J. Smith and S.-F. Chang. WebSEEK: a content-based image and video search and catalog tool for the Web. In *Proc. of IEEE Multimedia*, 1997.

[12] O. Zamir and O. Etzioni. Web document clustering: a feasibility demonstration. In *Proc. of the ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pp. 46-54, 1998.

[13] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma. Learning to cluster web search results. In *Proc. of the ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pp. 210-217, 2004.