BU CS TR98-004. To appear in **IEEE Workshop on Content-based Access of Image and Video Libraries**, June 1998.

Combining Textual and Visual Cues for Content-based Image Retrieval on the World Wide Web

Marco La Cascia, Saratendu Sethi, and Stan Sclaroff
Image and Video Computing Group
Computer Science Department
Boston University
Boston, MA 02215

Abstract

A system is proposed that combines textual and visual statistics in a single index vector for content-based search of a WWW image database. Textual statistics are captured in vector form using latent semantic indexing (LSI) based on text in the containing HTML document. Visual statistics are captured in vector form using color and orientation histograms. By using an integrated approach, it becomes possible to take advantage of possible statistical couplings between the content of the document (latent semantic content) and the contents of images (visual statistics). The combined approach allows improved performance in conducting content-based search. Search performance experiments are reported for a database containing 100,000 images collected from the WWW.

1 Introduction

The growing importance of the world wide web has led to the birth of a number of image search engines [6, 7, 11, 12]. The web's staggering scale puts severe limitations on the types of indexing algorithms that can be employed. Luckily, due to the scale and unstructured nature of the WWW, even the most basic indexing tools would be welcome. Existing image engines allow users to search for images via an SQL keywords interface [6, 12] and/or via query by image example (QBE) [7, 11, 12].

In QBE, the system presents an initial page of representative (or randomly selected) images thumbnails to the user [5]. The user then marks one or more images as relevant to the search. The visual statistics for these images are then used in defining a query. The user's success in locating images in the database depends in great part on which images appear within this initial group of thumbnails. Given the numerous and diverse images available in an WWW index, it is difficult to guarantee that there will be even one relevant image shown in the initial page. We call this the *page zero problem*.

Other WWW image engines allow the user to form a query in terms of SQL keywords [6, 12]. This alleviates the page zero problem, since the user can give text information that narrows the scope of possible images displayed on page zero. To build the image index, keywords are extracted heuristically from HTML documents containing each image, and/or from the image URL. Unfortunately, it is difficult to include visual cues within an SQL framework. This results in systems that force inherently visual information into a textual form, or systems that treat textual and visual cues disjointly.

2 New Approach

By truly unifying textual and visual statistics, one would expect to get better results than either used separately. In this paper, we propose an approach that allows the combination of visual statistics with textual statistics in the vector space representation commonly used in query by image content systems. Text statistics are captured in vector form using latent semantic indexing (LSI) [3].

Text documents are represented by low-dimensional vectors that can be matched against user queries in the LSI “semantic” space. The LSI index for an HTML document is then associated with each of the images contained therein. Visual statistics (e.g., color, orientedness) are also computed for each image. The LSI vector and visual statistics vector are then combined into a unified vector that can be used for content-based search of the resulting image database.

In previous systems, visual information has been used in a form that is compatible with standard DBMS frameworks [6] or only to refine the query [12]. By using an integrated approach, we are able to take advantage of possible statistical couplings between the content of the document (latent semantic content) and the contents of images (image statistics). Furthermore, LSI implicitly addresses problems with synonyms, word sense, lexical matching, and term omission.

HTML Tags	Weights
ALT field of IMG	6.00
TITLE	5.00
H1	4.00
H2	3.60
H3	3.35
H4	2.40
H5	2.30
H6	2.20
B	3.00
EM	2.70
I	2.70
STRONG	2.50
< No Tag >	1.00

Table 1: Word weights based on HTML tags.

3 Latent Semantic Indexing

The context in which an image appears can be abstracted from the containing HTML document using a method known in the information retrieval community as Latent Semantic Indexing (LSI)[3]. LSI works by statistically associating related words to the conceptual context of the given document. This structure is estimated by a truncated singular value decomposition (SVD).

To begin with, each HTML document is parsed and a word frequency histogram is computed. The documents in the database are not similar in length and structure. Also all words in the same HTML document may not be equally relevant to the document context. Hence words appearing with specific HTML tags are given special importance by assigning a higher weight as compared to all other words in the document.

The system assigns different weights to the words appearing in the *title, headers* and in the *alt* fields of the *img* tags along with words emphasized with different fonts like *bold, italics* etc. (see Table 1). These weight values have been fixed according to the likelihood of useful information that may be implied by the text. Weighting selectively the words appearing between various HTML tags helps in emphasizing the underlying information of that document. A different weighting scheme was used in [6].

In addition, words appearing before and after a particular image are also assigned a weight based upon their proximity to the image. The weighting value is computed as $\rho * e^{-2.0*pos/dist}$, where pos is the position of the word with respect to the image and $dist$ is the maximum number of words considered to apply such weighting. In the current implementation, the $dist$ is 10 and 20 for words appearing before and after the image respectively. ρ was fixed to be 5.0 so that the words nearest to the images get weighted slightly less than the words appearing in the *alt* field of that image and the *title* of the URL. Thus images appearing at different locations in an HTML document will have different LSI indices.

A term × image matrix A is created; the element a_{ij} represents the frequency of term i in the document containing image j with a weight based on its status and position with respect to the image. Retrieval may become biased if a term appears several times or never appears in a document. Hence further local and global weights may be applied to increase/decrease the importance of a term in and amongst documents. The element a_{ij} is expressed as the product of the local ($L(i,j)$) and the global weight ($G(i)$). Several weighting schemes have been suggested in the literature. Based on the performance reported in [4], the *log-entropy* scheme was chosen. According to this weighting scheme, the local and the global weights are given as below:

$$\begin{aligned} a_{ij}^{'} &= L(i,j) \times G(i) && (1) \\ L(i,j) &= log(a_{ij}+1) && (2) \\ G(i) &= 1 - \sum_k \frac{p_{ik} log(p_{ik})}{log(ndocs)} && (3) \\ p_{ik} &= tf_{ik} / \sum_k tf_{ik} && (4) \end{aligned}$$

where tf_{ik} is the *pure* term frequency for term i in HTML document k not weighted according to any scheme and $ndocs$ is the number of documents used in the training set.

The matrix A is then factored into U, Σ, V matrices using the singular value decomposition, $A = U \Sigma V^T$, where $U^T U = V^T V = I_n$ and $\Sigma = diag(\sigma_1, \cdots, \sigma_n), \sigma_i > 0$ for $1 \leq i \leq r$, $\sigma_j = 0$ for $j \geq r+1$. The columns of U and V are referred to as the left and right singular vectors respectively, and the diagonal elements of Σ are the singular values of A. The first r columns of the orthogonal matrices U and V define orthonormal eigenvectors associated with the r nonzero eigenvalues of AA^T and $A^T A$ respectively. For further details about SVD and the information conveyed by the matrices, readers are directed to [1].

The SVD decomposes the original term-image relationships into a set of linearly independent vectors. The dimension of the problem is reduced by choosing k most significant dimensions from the factor space which are then used for estimating the original index vectors. Thus SVD derives a set of uncorrelated indexing factors, whereby each term or image is represented as a vector in the k-space:

$$q^{'} = q^T U_k \Sigma_k^{-1}. \qquad (5)$$

The resulting LSI vector provides the context associated with an image and is combined with its computed visual feature vectors and stored in the database index.

4 Integration with Visual Statistics

The visual statistics we use to describe an image are the color histogram and dominant orientation histogram [11].

Given the potential number of images to index it is of fundamental importance that the dimension of the feature vector is as small as possible. As pointed out by White and Jain [13], the visual data has an *intrinsic dimension* that is significantly smaller than the original dimension. The use of principal component analysis (PCA) for each subvector space (color and directionality) allows us to dramatically reduce the dimension of the visual features with a small loss of information [11].

The global feature vector, representing the content of the image, is then composite of several subvectors: a dimensionally reduced color histogram and orientation histogram for each of 6 overlapping image regions [11], the LSI descriptor as described above. As each image is described by a vector, the query by example problem can be easily formulated as a k-nearest neighbor one. Our approach is slightly different. We allow the user to query the system based on several examples and use the additional information coming from the multiple selection to allow relevance feedback.

5 Relevance Feedback

Relevance feedback enables the user to iteratively refine a query via the specification of relevant items[10]. By including the user in the loop, better search performance can be achieved. Typically, the system returns a set of possible matches, and the user gives feedback by marking items as relevant or not relevant.

Given user-specified relevant images, the system must then infer what combination of measures should be used. The ImageRover system employs a relevance feedback algorithm that selects appropriate L_m Minkowski distance metrics on the fly. The algorithm determines the relative weightings of the individual features based on feedback images. This weighting thus varies depending upon the particular selections of the user. Due to space limitations, readers are referred to [11] for a complete description of our relevance feedback algorithm.

6 System Implementation

We implemented a fully functional system to test our technique. The web robots, at point of this writing, collected a few hundred thousand documents containing more than 250,000 unique[1] and valid images[2]. In practice some of the documents and the images are duplicated as sometimes the same document or the same image appears with a different URL due to name aliasing. To have a significant sampling of the images present on the WWW a list of links related to more diverse topics is needed. We selected the Yahoo pages reported in Table 2 as the starting point of our web robots. This version of the system is available on-line[3] and the reader is encouraged to try it.

Yahoo category
Science:Astronomy:Pictures
Science:Biology:Zoology:Animals:Pictures
Recreation:Aviation:Pictures
Arts:Visual Arts:Photography:Underwater
Arts:Visual Arts:Photography:Photographers
Arts:Visual Arts:Photography:Nature and Wildlife
Recreation:Travel:Pictures
Computers and Internet:Multimedia:Pictures
Recreation:Sports:News and Media:Magazines
Companies:Arts and Crafts:Galleries
News and Media:Television:Cable:Networks:US

Table 2: Starting points for web robot image collection.

The user interacts with the system through a web browser. To get the search going, a set of starting images has to be shown to the user. In our system, this problem can be handled relying on the LSI information associated with the images in our database. In fact, the user specifies a set of keywords; this set of keywords can be considered as a text document. An LSI index is computed for the keywords and used to match nearest neighbors in the subspace of all LSI vectors in our image database.

Once the user finds and marks one or more images to guide the search, the user can initiate a query with a click on the search button. Similar images (the number of returned images is a user chosen value) are then retrieved and shown to the user in decreasing similarity order. The user can then select other relevant images to guide next search and/or deselect one or more of the query images and iterate the query. There is no limit on the number of feedback iterations nor on the number of example images employed.

The current implementation of the system is not optimized for speed. The query server and the web server run on an SGI Origin 200 with 4 R10000 180 MHz processors and 1 GB RAM. As all the data can be kept in memory[4] a brute force search of the k nearest neighbors takes less than 1 second. In the case of the page zero we have to compute the LSI vector corresponding to the keywords provided by the user. This is a simple vector by matrix product, and, even though the dimension is high, it takes again less than one second. We use the NCSA Apache web server to display the thumbnails of the retrieved images.

7 Experimental Evaluation

We tested our system with human subjects. Our experiments were intended to evaluate the effectiveness of per-

[1] We consider two images different if they have different URLs.

[2] By valid image we mean an image that has both its width and height greater than 64 pixels; images not satisfying this heuristic are discarded.

[3] `http://www.cs.bu.edu/groups/ivc/ImageRover`

[4] After the dimensionality reduction the visual features are represented by around 200 dimensional vector, so less than 500 floating point number (about 2 KB) per image are required and keeping in memory 100,000 images requires 200 MB.

formance achieved through the combined use of visual and textual features.

The system was initially trained using a randomly selected sub-set of the collected data containing 58,908 images. The eigendecompositions for the LSI vectors and the visual features are performed only once and new images are inserted into the system only as a projection into the reduced feature spaces. It may be pointed out that since the training set was selected so that it is representative of the documents available on the web and since the size of training set is considerable, it may not require to retrain the system. Standard techniques have been reported in [2] for updating SVD-based indexing schemes.

After the training step, we indexed a set of 10,000 images disjoint with the training set and stored them in the database. A subset of 100 images in the database was selected (randomly) to be retrieved. Two subjects were asked to find each of the images using our system, one at a time. The subjects were first presented with one of the 100 images, displayed in a window on the computer screen. While the target image remained visible, the subjects could then formulate a query of the WWW image database via the user interface as described above. This process was repeated for all 100 images in the test set.

In practice, for each image, the subjects independently typed in a few keywords relevant to the target image to obtain a starting set of 100 images (page zero). In the current experiments the subjects have used four keywords on average per search. The subjects could then further refine the search through iterations of the relevance feedback mechanism. At each iteration, the system displayed the 100 top matches for the query.

A search was considered successful if the subjects could get the target image displayed in the top 100. A search was considered unsuccessful if the subject could not get the target image displayed in the top 100 within four iterations of relevance feedback. This limit on the number of feedback iterations was chosen to reflect the amount of time a typical user would be willing to devote in finding an image.

To measure how the system scales with the number of images archived, the experiments were repeated at various database sizes: 10,000, 30,000, 50,000 and 100,000 images respectively. To evaluate search performance with respect to the types of feature vectors employed, multiple trials were conducted in which textual only, visual only, and combined textual and visual features were included in the indexing vector. In each set of trials, subjects were asked to find the same randomly selected subset of 100 images. To avoid biasing of the subjects, due to an increased familiarity with the data set, we asked them to use the same keywords in generating page zero during each trial.

The average percentage of target images that subjects were able to successfully retrieve, is shown in Figure 1. The graph depicts percentage of images found as a function of the number of images contained in the test database. The percentage of images that the subjects were able to retrieve decreases as the number of images increases but is still reasonable considering the database size. With a database size of 10,000 the retrieval success rate with combined cues was around 35%, degrading to about 18% as the database size increased to 100,000.

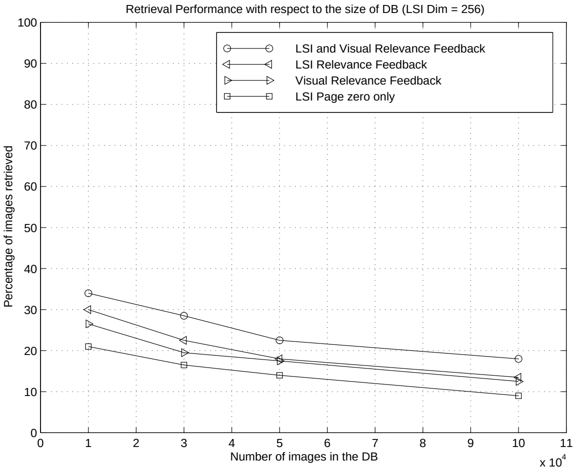

Figure 1: Percentage of test images retrieved vs. # of images in the database. LSI and visual information were both used in our integrated relevance feedback framework.

The lowest curve on the graph shows the percentage of images that subjects retrieved in page zero. The other curves show the performance when users were allowed to use relevance feedback. To determine the major contributor in performance improvement, experimental trials were conducted in which system employed visual statistics only, LSI only, and LSI and visual statistics combined in relevance feedback. As can be seen, relevance feedback using combined features offers a significant performance improvement over using either of the features separately.

In a separate set of trials with a different test set, the sensitivity to LSI dimension was tested for a database of 10,000 images at six levels: $dim(LSI) = 64, 128, 256, 384, 768$. The tests were done for searching 50 random images. Results of this experiment are shown in Fig.2. The graph shows how subject's success rates improved when a higher LSI dimension was employed. The steepest improvement in performance was achieved with LSI dimension of 256. After that, performance increased more slowly with increasing LSI dimension. This is consistent with results reported in [4], where it was observed that as LSI dimension increases the performance curve flattens out, and then actually drops off slightly (due to noise).

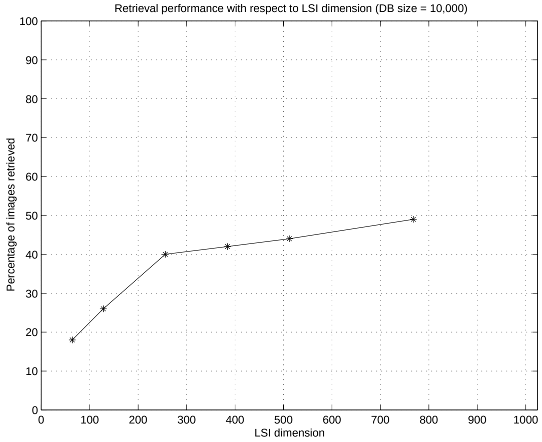

Figure 2: Search performance with respect to LSI dimension. Relevance feedback was determined using combined visual and textual cues.

8 Discussion

It is evident from our experiments that based on text features only, it is sometimes possible for subjects to find the target image in page zero. If the target image did not appear in page zero, then the subjects were twice as likely to find that image when textual and visual features were combined in relevance feedback.

In the experiments, it was observed that the average number of relevance feedback steps required to steer the system towards the wanted image was independent of the database size and LSI dimensions used. Whenever an image was found using relevance feedback it was found in 1.9 feedback iterations on average.

We performed an analysis to characterize why subjects were unable to retrieve certain images from the index. We observed that often, the inability to find a particular image is due to a lack of correlation between the image content and the surrounding text; *e.g.,* banners, specific logos, etc. It was also found that in some cases, there are web pages like photo galleries which contain several images and/or very little text. Finally, in some cases, subjects were simply unable to form a page zero query, because it was difficult to describe the content of a particular test image with words.

We experimentally found that a 256-dimensional LSI vector leads to good results for our data set, despite the breadth of the subject matter of documents included in the LSI training. The optimal LSI dimension may also be obtained using an MDL framework [14]. Figure 2 shows that the net increase in the retrieval performance drops significantly after 256 LSI dimension.

We expect that even in a very large database (with millions of images) using our technique it will be possible to retrieve specific images. In other cases the user will be able to find several relevant images. To gain better search accuracy, we are investigating the use of modular eigenspaces [8, 9] for modeling various LSI subject categories.

In summary, the maximum improvement was achieved when both visual and textual information were used in the relevance feedback framework. The experiments show that this improvement is significantly larger than that achievable using visual only or textual only information in the query refinement phase.

Acknowledgments

This work is sponsored in part through grants from the National Science Foundation: Faculty Early Career Award #IIS-9624168, CISE Research Infrastructure Awards #CDA-9623865 and #CDA-9529403, and the PACI program #ACI-9619019.

References

[1] M. Berry and S Dumais. Using linear algebra for intelligent information retrieval. TR UT-CS-94-270, U. Tenn., 1994.

[2] G. Brien. Information Management Tools for Updating an SVD-Encoded Indexing Scheme. TR UT-CS-94-259, U. Tenn., 1994.

[3] S. Deerwester, S. Dumais, T. Landauer, G. Furnas, and R. Harshman. Indexing by latent semantic analysis. *J. of the Soc. for Info. Sci.*, 41(6):391–407, 1990.

[4] S. Dumais. Improving the retrieval of information from external sources. *Behavior Res. Meth., Instruments and Comp.*, 23(2):229–236, 1991.

[5] M. Flickner et al. Query by image and video content: the QBIC system. *IEEE Computer*, pp. 23–30, Sep. 1995.

[6] C. Frankel, M. Swain, and V. Athitsos. Webseer: An image search engine for the world wide web. TR 96-14, U. Chicago, 1996.

[7] T. Gevers and A. Smeulders. Pictoseek: A content-based image search engine for the www. *Proc. Int. Conf. on Visual Info.*, Dec. 1997.

[8] H. Murase and S. Nayar. Visual Learning and Recognition of 3-D Objects from Appearance. *IJCV*, 14(1):5–24, 1995.

[9] A. Pentland, B. Moghaddam, T. Starner, O. Oliyide, and M. Turk. View-based and modular eigenspaces for face recognition. *Proc. CVPR*, pp. 84–91, 1994.

[10] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1989.

[11] S. Sclaroff, L. Taycher, and M. La Cascia. ImageRover: A content-based image browser for the world wide web. *Proc. of IEEE Int. Workshop on Content-Based Access of Image and Video Libraries*, 1997.

[12] J. Smith and S. Chang. Visually searching the web for content. *IEEE Multimedia*, 4(3):12–20, July 1997.

[13] D.A. White and R. Jain. Algorithms and strategies for similarity retrieval. TR VCL-96-101, UCSD, 1996.

[14] Hongyuan Zha. A subspace-based model for information retrieval with applications in latent semantic indexing. TR CSE-98-002, Penn State, 1998.