Content-Based Image Retrieval by Clustering

Yixin Chen*
Dept. of Computer Science
Univ. of New Orleans
New Orleans, LA 70148
yixin@cs.uno.edu

James Z. Wang
School of IST
The Penn. State Univ.
University Park, PA 16802
jwang@ist.psu.edu

Robert Krovetz
Teoma Technologies
1151 S Washington Ave, 400
Piscataway, NJ 08554
krovetz@nec-labs.com

ABSTRACT

In a typical content-based image retrieval (CBIR) system, query results are a set of images sorted by feature similarities with respect to the query. However, images with high feature similarities to the query may be very different from the query in terms of semantics. This is known as the semantic gap. We introduce a novel image retrieval scheme, CLUster-based rEtrieval of images by unsupervised learning (CLUE), which tackles the semantic gap problem based on a hypothesis: *semantically similar images tend to be clustered in some feature space*. CLUE attempts to capture semantic concepts by learning the way that images of the same semantics are similar and retrieving image clusters instead of a set of ordered images. Clustering in CLUE is dynamic. In particular, clusters formed depend on which images are retrieved in response to the query. Therefore, the clusters give the algorithm as well as the users semantic relevant *clues* as to where to navigate. CLUE is a general approach that can be combined with any real-valued symmetric similarity measure (metric or nonmetric). Thus it may be embedded in many current CBIR systems. Experimental results based on a database of about 60,000 images from COREL demonstrate improved performance.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: clustering; query formulation; retrieval models; search process.

General Terms

Algorithms, Design, Experimentation, Human Factors.

Keywords

Content-based image retrieval, image classification, unsupervised learning, spectral graph clustering.

*Y. Chen was with Department of Computer Science and Engineering, The Pennsylvania State University.

MIR'03, November 7, 2003, Berkeley, California, USA.

1. INTRODUCTION

The steady growth of the Internet, the falling price of storage devices, and an increasing pool of available computing power make it necessary and possible to manipulate very large repository of digital information efficiently. Generally speaking, content-based image retrieval (CBIR) aims at developing techniques that support effective searching and browsing of large image digital libraries based on automatically derived image features. Although CBIR is still immature, there has been abundance of prior work. Due to space limitations, we only review work most related to ours, which by no means represents the comprehensive list.

1.1 Previous Work

In the past decade, many general-purpose image retrieval systems have been developed. Examples include QBIC System [6], Photobook System [16], Blobworld System [3], Virage System [9], VisualSEEK and WebSEEK Systems [20], the PicHunter System [5], NeTra System [14], MARS System [15], and SIMPLIcity Systems [22].

A typical CBIR system views the query image and images in the database (target images) as a collection of features, and ranks the relevance between the query image and any target images in proportion to feature similarities. Nonetheless, the meaning of an image is rarely self-evident. Images with high feature similarities to the query image may be very different from the query in terms of the interpretation made by a user (*user semantics* or, in short, *semantics*). This is referred to as the *semantic gap*, which reflects the discrepancy between the relatively limited descriptive power of low level imagery features and the richness of user semantics.

Depending on the degree of user involvement in the retrieval process, generally, two classes of approaches have been proposed to reduce the semantic gap: relevance feedback and image database preprocessing using statistical classification. A relevance-feedback-based approach allows a user to interact with the retrieval algorithm by providing the information of which images he or she thinks are relevant to the query [5, 17]. Based on the user feedbacks, the model of similarity measure is dynamically updated to give a better approximation of the perception subjectivity. Empirical results demonstrate the effectiveness of relevance feedback for certain applications. Nonetheless such a system may add burden to a user especially when more information is required than just Boolean feedback (relevant or non-relevant).

Statistical classification methods group images into semantically meaningful categories using low level visual fea-

tures so that semantically-adaptive searching methods applicable to each category can be applied [18, 21, 22, 12]. For example, SemQuery system [18] categorizes images into different set of clusters based on their heterogeneous features. Vailaya et al. [21] organize vacation images into a hierarchical structure. At the top level, images are classified as indoor or outdoor. Outdoor images are then classified as city or landscape that is further divided into sunset, forest, and mountain classes. The SIMPLIcity system [22] classifies images into graph, textured photograph, or non-textured photograph, and thus narrows down the searching space in a database. ALIP system [12] uses categorized images to train hundreds of two-dimensional multiresolution hidden Markov models each corresponding to a semantic category. Although these classification methods are successful in their specific domains of application, the simple ontology built upon them could not incorporate the rich semantics of a sizable image database. There has been work on attaching words to images by associating the regions of an image with object names based on region-term co-occurrence [2]. But as noted by the authors in [2], the algorithm relies on semantically meaningful segmentation. And semantically precise image segmentation by an algorithm is still an open problem in computer vision [19, 23].

1.2 Motivation

Figure 1 shows a query image and the top 29 target images returned by a CBIR system described in [4] where the query image is on the upper-left corner. From left to right and top to bottom, the target images are ranked according to decreasing values of similarity measure. In essence, this can be viewed as a one-dimensional visualization of the image database in the "neighborhood" of the query image using a similarity measure. If the query image and majority of the images in the "vicinity" have the same semantics, then we would expect good results. But target images with high feature similarities to the query image may be quite different from the query image in terms of semantics due to the semantic gap. For the example in Figure 1, the target images belong to several semantic classes where the dominant ones include horses (11 out of 29), flowers (7 out of 29), golf player (4 out of 29), and vehicle (2 out of 29).

However, the majority of top matches in Figure 1 belong to a quite small number of distinct semantic classes, which suggests a hypothesis that, in the "vicinity" of the query image, images of the same semantics are more similar to each other than to images of different semantics. Or, in other words, images tend to be semantically clustered. Therefore, a retrieval method, which is capable of capturing this structural relationship, may render semantically more meaningful results to the user than merely a list of images sorted by a similarity measure. Similar hypothesis has been well studied in document (or text) retrieval [1] where strong supporting evidence has been presented [10].

This motivates us to tackle the semantic gap problem from the perspective of unsupervised learning. In this paper, we propose an algorithm, CLUster-based rEtrieval of images by unsupervised learning (CLUE), to retrieve image clusters instead of a set of ordered images: the query image and neighboring target images, which are selected according to a similarity measure, are clustered by an unsupervised learning method and returned to the user. In this way, relations among retrieved images are taken into consideration through clustering and may provide extra information for ranking and presentation. CLUE has the following characteristics:

- It is a cluster-based image retrieval scheme that can be used as an alternative to retrieving a set of ordered images. The image clusters are obtained from an unsupervised learning process based on not only the feature similarity of images to the query, but also how images are similar to each other. In this sense, CLUE aims to capture the underlying concepts about how images of the same semantics are alike and present to the users semantic relevant *clues* as to where to navigate.

- It is a similarity-driven approach that can be built upon virtually any symmetric real-valued image similarity measure. Consequently, our approach could be combined with many other image retrieval schemes including the relevance feedback approach with dynamically updated models of similarity measure.

- It provides a dynamic and local visualization of the image database using a clustering technique. The clusters are created depending on which images are retrieved in response to the query. Consequently, the clusters have the potential to be closely adapted to characteristics of a query image. Moreover, by constraining the collection of retrieved images to the neighborhood of the query image, clusters generated by CLUE provides a local approximation of the semantic structure of the whole image database. Although the overall semantic structure of the database could be very complex and extremely difficult to identify by a computer program, locally it may be well described by a simple approximation such as clusters. This is in contrast to current image database statistical classification methods [18, 21, 22], in which the semantic categories are derived for the whole database in a preprocessing stage, and therefore are global, static, and independent of the query.

1.3 Outline of the Paper

The remainder of the paper is organized as follows. Section 2 describes the general methodology of CLUE. Section 3 provides the experimental results. We conclude in Section 4, together with a discussion of future work.

2. RETRIEVAL OF IMAGE CLUSTERS

2.1 System Overview

For the purpose of simplifying the explanations, we call a CBIR system using CLUE a Content-Based Image Clusters Retrieval (CBICR) system. From a data-flow viewpoint, a general CBICR system can be characterized by the diagram in Figure 2. The retrieval process starts with feature extraction for a query image. The features for target images (images in the database) are usually precomputed and stored as feature files. Using these features together with an image similarity measure, the resemblance between the query image and target images are evaluated and sorted. Next, a collection of target images that are "close" to the query image are selected as the neighborhood of the query image. A clustering algorithm is then applied to these target images. Finally, the system displays the image clusters and adjusts the similarity model according to user feedback (if relevance feedback is included).

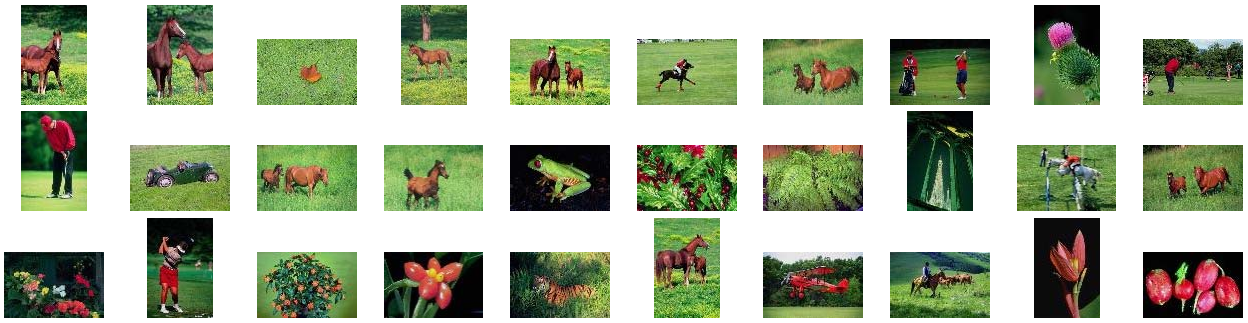

Figure 1: A query image and its top 29 matches returned by the CBIR system at `http://wang.ist.psu.edu/IMAGE` (UFM). The query image is on the upper-left corner. The ID number of the query image is 6275.

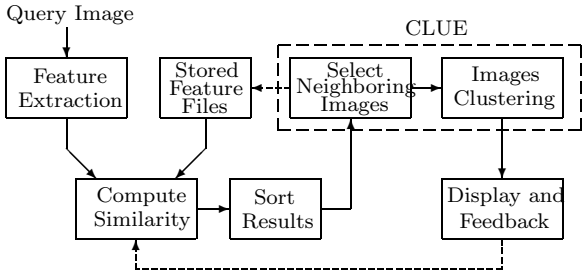

Figure 2: A diagram of a general CBICR system. The arrows with dotted lines may not exist for some CBICR systems.

The major difference between CBICR and CBIR systems lies in the two processing stages, selecting neighboring target images and image clustering, which are the major components of CLUE. A typical CBIR system bypasses these two stages and directly outputs the sorted results to the display and feedback stage. Figure 2 suggests that CLUE can be designed independent of the rest of the components because the only information needed by CLUE is the sorted similarities. This implies that CLUE may be embedded in a typical CBIR system regardless of the image features being used, the sorting method, and whether there is feedback or not. As a result, in the following subsections, we focus on the discussion of general methodology of CLUE, and assume that a similarity measure is given.

2.2 Neighboring Target Images Selection

To mathematically define the neighborhood of a point, we need to first choose a measure of distance. As to images, the distance can be defined by either a similarity measure (a larger value indicates a smaller distance) or a dissimilarity measure (a smaller value indicates a smaller distance). Because simple algebraic operations can convert a similarity measure into a dissimilarity measure, without loss of generality, we assume that the distance between two images is determined by a symmetric dissimilarity measure, $d(i, j) = d(j, i) \geq 0$, and name $d(i, j)$ the distance between images i and j to simplify the notation.

Next we propose two simple methods to select a collection of neighboring target images for a query image i:

1. *Fixed radius method* (FRM) takes all target images within some fixed radius ϵ with respect to i. For a given query image, the number of neighboring target images is determined by ϵ.

2. *Nearest neighbors method* (NNM) first chooses k nearest neighbors of i as seeds. The r nearest neighbors for each seed are then found. Finally, the neighboring target images are selected to be all the distinct target images among seeds and their r nearest neighbors.

If the distance is metric, both methods will generate similar results under proper parameters (ϵ, k, and r). However, for non-metric distances, especially when the triangle inequality is not satisfied, the set of target images selected by two methods could be quite different regardless of the parameters. This is due to the violation of the triangle inequality: the distance between two images could be huge even if both of them are very close to a query image. The NNM is used in this work. Compared with the FRM, our empirical results show that, with proper choices of k and r, the NNM tends to generate more structured collection of target images under a non-metric distance. On the other hand, the computational cost of the NNM is higher than that of the FRM because of the extra time to find nearest neighbors for all k seeds. The time complexity can be reduced at the price of extra storage space.

2.3 Weighted Graph Representation and Spectral Graph Partitioning

Data representation is typically the first step to solve any clustering problem. In the field of computer vision, two types of representations are widely used. One is called the *geometric representation*, in which data items are mapped to some real normed vector space. The other is the *graph representation*. It emphasizes the pairwise relationship, but is usually short of geometric interpretation. When working with images, the geometric representation has a major limitation: it requires that the images be mapped to points in some real normed vector space. Overall, this is a very restrictive constraint. For example, in region-based algorithms [4, 13, 22], an image is often viewed as a collection of regions. The number of regions may vary among images. Although regions can be mapped to certain real normed vector space, it is in general impossible to do so for images unless the distance between images is metric, in which case embedding becomes feasible. Nevertheless, many distances for images are non-metric for reasons given in [11].

Therefore, this paper adopts a graph representation of

neighboring target images. A set of n images is represented by a weighted undirected graph $G = (\mathbf{V}, \mathbf{E})$: the nodes $\mathbf{V} = \{1, 2, \ldots, n\}$ represent images, the edges $\mathbf{E} = \{(i, j) : i, j \in \mathbf{V}\}$ are formed between every pair of nodes, and the non-negative weight w_{ij} of an edge (i, j), indicating the similarity between two nodes, is a function of the distance (or similarity) between nodes (images) i and j. Given a distance $d(i, j)$ between images i and j, we define $w_{ij} = e^{-\frac{d(i,j)^2}{s^2}}$ where s is a scaling parameter that needs to be tuned to get suitable locality property. The weights can be organized into a matrix $\mathbf{W}$, named the *affinity matrix*, with the ij-th entry given by w_{ij}. Although it is a relatively simple weighting scheme, our experimental results (Section 3) have shown its effectiveness. The same scheme has been used in [8, 19]. Supports for exponential decay from psychological studies are also provided by [8].

Under a graph representation, clustering can be naturally formulated as a graph partitioning problem. Among many graph-theoretic algorithms, this paper uses the normalized cut (Ncut) algorithm [19] for image clustering. Roughly speaking, Ncut method attempts to organize nodes into groups so that the within-group similarity is high, and/or the between-groups similarity is low. Compared with many other spectral graph partitioning methods, such as average cut and average association, the Ncut method is empirically shown to be relatively robust in image segmentation [19]. The Ncut method can be recursively applied to get more than two clusters. But this leads to the questions: 1) which subgraph should be divided? and 2) when should the process stop? In this paper, we use a simple heuristic. Each time the subgraph with the maximum number of nodes is partitioned (random selection for tie breaking). The process terminates when the bound on the number of clusters is reached or the Ncut value exceeds some threshold T.

2.4 Finding Representative Images

Ultimately, the system needs to present the image clusters to the user. Unlike a typical CBIR system, which displays certain numbers of top matched target images to the user, a CBICR system should be able to provide an intuitive visualization of the clustered structure in addition to all the retrieved target images. For this reason, we propose a two-level display scheme. At the first level, the system shows a collection of representative images of all the clusters (one for each cluster). At the second level, the system displays all target images within the cluster specified by a user.

Nonetheless two questions still remain: 1) how to organize these clusters? and 2) how to find a representative image for each cluster? The organization of clusters will be described in Section 2.5. For the second question, we define a representative image of a cluster to be the image that is most similar to all images in the cluster. This statement can be mathematically illustrated as follows. Given a graph representation of images $G = (\mathbf{V}, \mathbf{E})$ with affinity matrix $\mathbf{W}$, let the collection of image clusters $\{\mathbf{C}_1, \mathbf{C}_2, \cdots, \mathbf{C}_m\}$ be a partition of $\mathbf{V}$. The representative node (image) of $\mathbf{C}_i$ is

$$\arg\max_{j \in \mathbf{C}_i} \sum_{t \in \mathbf{C}_i} w_{jt} \ . \tag{1}$$

Basically, for each cluster, we pick the image that has the maximum sum of within cluster similarities.

2.5 Organization of Clusters

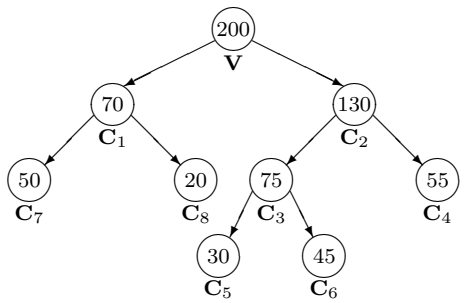

Figure 3: A tree generated by four Ncuts on V. The numbers denote the size of the corresponding clusters.

The recursive Ncut partition is essentially a hierarchical divisive clustering process that produces a tree. For example, Figure 3 shows a tree generated by four recursive Ncuts. The first Ncut divides $\mathbf{V}$ into $\mathbf{C}_1$ and $\mathbf{C}_2$. Since $\mathbf{C}_2$ has more nodes than $\mathbf{C}_1$, the second Ncut partitions $\mathbf{C}_2$ into $\mathbf{C}_3$ and $\mathbf{C}_4$. Next, $\mathbf{C}_3$ is further divided because it is larger than $\mathbf{C}_1$ and $\mathbf{C}_4$. The fourth Ncut is applied to $\mathbf{C}_1$, and gives the final five clusters (or leaves): $\mathbf{C}_4$, $\mathbf{C}_5$, $\mathbf{C}_6$, $\mathbf{C}_7$, and $\mathbf{C}_8$.

The above example suggests trees as a natural organization of clusters. Nonetheless, the tree organization here may be misleading to a user because there is no guarantee of any correspondence between the tree and the semantic structure of images. Furthermore, organizing image clusters into a tree structure will significantly complicate the user interface. So, in this work, we employ a simple linear organization of clusters called *traversal ordering*: arrange the leaves in the order of a binary tree traversal (left child goes first). The order of two clusters produced by an Ncut iteration is decided by an arbitration rule: 1) let $\mathbf{C}_1$ and $\mathbf{C}_2$ be two clusters generated by an Ncut on $\mathbf{C}$, and d_1 (d_2) be the minimal distance between the query image and all images in $\mathbf{C}_1$ ($\mathbf{C}_2$); 2) if $d_1 < d_2$ then $\mathbf{C}_1$ is the left child of $\mathbf{C}$, otherwise, $\mathbf{C}_2$ is the left child. Under the traversal ordering and arbitration rule, the query image is in the leftmost leaf ($\mathbf{C}_7$ in Figure 3) since a cluster containing the query image will have a minimum distance (d_1 or d_2) of 0, and thus will always be assigned to the left child. For the sake of consistency, images within each cluster are also organized in ascending order of distances to a query image.

3. EXPERIMENTS

3.1 User Interface

Our experimental CBICR system uses the same feature extraction scheme and UFM similarity measure as those in [4]. The system is implemented with a general-purpose image database (from COREL), which includes about 60,000 images. The system has a very simple CGI-based query interface. It provides a *Random* option that will give a user a random set of images from the image database to start with. In addition, users can either enter the ID of an image as the query or submit any image on the Internet as a query by entering the URL of the image. Once a query image is received, the system displays a list of thumbnails

each of which represents an image cluster. The thumbnails are found according to (1), and sorted using the algorithm described in Section 2.5. A user can view all images in the associated cluster by clicking a thumbnail.

3.2 Query Examples

To qualitatively evaluate the performance of the system over the 60,000-image COREL database, we randomly pick five query images with different semantics, namely, *birds*, *car*, *food*, *historical buildings*, and *soccer game*. For each query example, we examine the precision of the query results depending on the relevance of the image semantics. Here only images in the first cluster, in which the query image resides, are considered. This is because images in the first cluster can be viewed as sharing the same similarity-induced semantics as that of the query image according to the clusters organization described in Section 2.5. Performance issues about the rest clusters will be covered in Section 3.3. Since CLUE is built upon UFM similarity measure, query results of a typical CBIR system, SIMPLIcity system using UFM similarity measure [4] (we call the system UFM to simplify notation), are also included for comparison. We admit that the relevance of image semantics depends on standpoint of a user. Therefore, our relevance criteria, specified in Figure 4, may be quite different from those used by a user of the system. Due to space limitations, only the top 11 matches to each query are shown in Figure 4. We also provide the number of relevant images in the first cluster (for CLUE) or among top 31 matches (for UFM).

Compared with UFM, CLUE provides semantically more precise results for the queries given in Figure 4. This is reasonable since CLUE utilizes more information about image similarities than UFM does. CLUE groups images into clusters based on pairwise distances so that the within-cluster similarity is high; and between-clusters similarity is low. The results seem to indicate that, to some extent, CLUE can group together semantically similar images.

3.3 Systematic Evaluation

To provide a more objective evaluation and comparison, CLUE is tested on a subset of the COREL database, formed by 10 image categories, each containing 100 images. The categories are *Africa*, *Beach*, *Buildings*, *Buses*, *Dinosaurs*, *Elephants*, *Flowers*, *Horses*, *Mountains*, and *Food* with corresponding Category IDs denoted by integers from 1 to 10, respectively. Within this database, it is known whether two images are of the same semantics. Therefore we can quantitatively evaluate and compare the performance of CLUE in terms of the goodness of image clustering and retrieval accuracy. In particular, the goodness of image clustering is measured via the distribution of images semantics in the cluster, and a retrieved image is considered a correct match if and only if it is the same category as the query image. These assumptions are reasonable since the 10 categories were chosen so that each depicts a distinct semantic topic.

3.3.1 Goodness of Image Clustering

Ideally, CLUE would be able to generate image clusters each of which contains images of similar or even identical semantics. The *confusion matrix* is one way to measure clustering performance. However, to compute the confusion matrix, the number of clusters needs to be equal to the number of distinct semantics, which is unknown in practice.

Table 1: Statistics of the average number of clusters m_i and the average cluster size v_i, and the correct categorization rate C_t.

ID. Category Name	Mean m_i	Mean $v_i \pm$ STDV	C_t
1. Africa	7.77	14.0 ± 3.80	0.75
2. Beach	7.96	13.6 ± 2.11	0.55
3. Buildings	7.89	11.8 ± 3.81	0.69
4. Buses	7.88	8.61 ± 3.49	0.88
5. Dinosaurs	7.96	6.51 ± 0.68	1.00
6. Elephants	7.52	14.6 ± 3.94	0.64
7. Flowers	8.00	8.84 ± 1.79	0.95
8. Horses	8.00	9.98 ± 2.95	0.97
9. Mountains	7.84	14.0 ± 2.70	0.51
10. Food	7.79	12.2 ± 2.48	0.78

Although we can force CLUE to always generate 10 clusters in this particular experiment, the experiment setup would then be quite different to a real application. So we use *purity* and *entropy* to measure the goodness of image clustering.

Assume we are given a set of n images belonging to c distinctive categories (or semantics) denoted by $1, \cdots, c$ (in this experiment $c \leq 10$ depending on the collection of images generated by NNM) while the images are grouped into m clusters $\mathbf{C}_j$, $j = 1, \cdots, m$. Purity for $\mathbf{C}_j$ is defined as

$$p(\mathbf{C}_j) = \frac{1}{|\mathbf{C}_j|} \max_{k=1,\cdots,c} |\mathbf{C}_{j,k}| \tag{2}$$

where $\mathbf{C}_{j,k}$ consists of images in $\mathbf{C}_j$ that belong to category k, and $|\mathbf{C}_j|$ represents the size of the set. Each cluster may contain images of different semantics. Purity gives the ratio of the dominant semantic class size in the cluster to the cluster size itself. The value of purity is always in the interval $[\frac{1}{c}, 1]$ with a larger value means that the cluster is a "purer" subset of the dominant semantic class. Entropy is another cluster quality measure, which is defined as follows:

$$h(\mathbf{C}_j) = -\frac{1}{\log c} \sum_{k=1}^{c} \frac{|\mathbf{C}_{j,k}|}{|\mathbf{C}_j|} \log \frac{|\mathbf{C}_{j,k}|}{|\mathbf{C}_j|} \ . \tag{3}$$

Since entropy considers the distribution of semantic classes in a cluster, it is a more comprehensive measure than purity. Note that we have normalized entropy so that the value is between 0 and 1. Contrary to the purity measure, an entropy value near 0 means the cluster is comprised mainly of 1 category, while an entropy value close to 1 implies that the cluster contains a uniform mixture of all categories.

The following are some additional notations used in the performance evaluation. For a query image i: 1) m_i denotes the *number of retrieved clusters*; 2) v_i is the *average size* of the retrieved clusters; 3) $P(i)$ is the *average purity* of the retrieved clusters, i.e., $P(i) = \frac{1}{m_i} \sum_{j=1}^{m_i} p(\mathbf{C}_j)$ where $p(\mathbf{C}_j)$ is computed according to (2); and 4) $H(i)$ is the *average entropy* of the retrieved clusters, i.e., $H(i) = \frac{1}{m_i} \sum_{j=1}^{m_i} h(\mathbf{C}_j)$ where $h(\mathbf{C}_j)$ is computed according to (3).

Every image in the 1000-image database is tested as a query. For query images within one semantic category, the following statistics are computed: the mean of m_i, the mean and standard deviation (STDV) of v_i, the mean of $P(i)$, and the mean of $H(i)$. In addition, we calculate P_{NNM} and H_{NNM} for each query, which are respectively the purity and entropy of the whole collection of images generated by NNM, and the mean of P_{NNM} and H_{NNM} for query images

Figure 4: Comparison of CLUE and UFM. The query image is the upper-left corner image of each block of images. The underlined numbers below the images are the ID numbers of the images in the database. For the images in the left column, the other number is the cluster ID (the image with a border around it is the representative image for the cluster). For images in the right column, the other two numbers are the value of UFM measure between the query image and the matched image, and the number of regions in the image. (a) birds, (b) car, (c) food, (d) historical buildings, and (e) soccer game.

within one semantic category. The results are summarized in Table 1 (second and third columns) and Figure 5. The third column of Table 1 shows that the size of clusters does not vary greatly within a category. This is because of the heuristic used in recursive Ncut: always dividing the largest cluster. It should be observed from Figure 5 that CLUE provides good quality clusters in the neighborhood of a query image. Compared with the purity and entropy of collections of images generated by NNM, the quality of the clusters generated by recursive Ncut is on average much improved for all categories except Category 5, for which NNM generates quite pure collections leaving little room for improvement.

3.3.2 *Retrieval Accuracy*

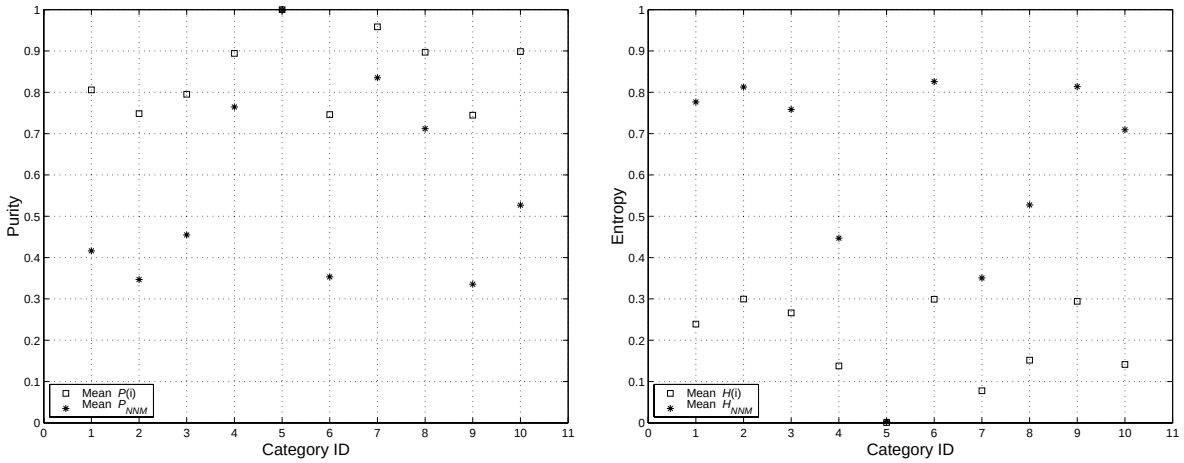

Figure 5: Purity and entropy of clusters. For mean $P(i)$ and mean P_{NNM}, larger numbers indicate purer clusters. For mean $H(i)$ and mean H_{NNM}, smaller numbers denote better cluster quality.

For image retrieval, purity and entropy by themselves may not provide a comprehensive estimate of the system performance even though they measure the quality of image clusters. Because what could happen is a collection of semantically pure image clusters but none of them sharing the same semantics with the query image. Therefore one needs to consider the semantic relationship between these image clusters and the query image. For this purpose, we introduce the *correct categorization rate* and *average precision*.

We call a query image being correctly categorized if the query category dominates the query image cluster. The correct categorization rate, C_t, for image category t is defined as the percentile of images in category t that are correctly categorized when used as queries. It indicates how likely the dominant semantics of the first cluster coincides with the query semantics. The fourth column of Table 1 lists estimations of C_t for 10 categories used in our experiments. Note that randomly assigning a dominant category to the query image cluster will give a C_t of value around 0.1.

From the standpoint of a system user, C_t may not be the most important performance index. Even if the first cluster, in which the query image resides, does not contain any images that are semantically similar to the query image, the user can still look into the rest clusters. So we use *precision* to measure how likely an user would find images belonging to the query category within a certain number of top matches. Here the precision is computed as the percentile of images belonging to the category of query image in the first 100 retrieved images. The *recall* equals precision for this special case since each category has 100 images. The r parameter in the NNM is set to be 30 to ensure that the number of neighboring images generated is greater than 100. As mentioned in Section 2.5, the linear organization of clusters may be viewed as a structured sorting of clusters in ascending order of distances to a query image. Therefore the top 100 retrieved images are found according to the order of clusters. The *average precision* for a category t is then defined as the mean of precisions for query images in category t. Figure 6 compares the average precisions given by CLUE with those obtained by UFM. Clearly, CLUE performs better than UFM for 9 out of 10 categories (they tie on the remaining one category). The overall average precisions for

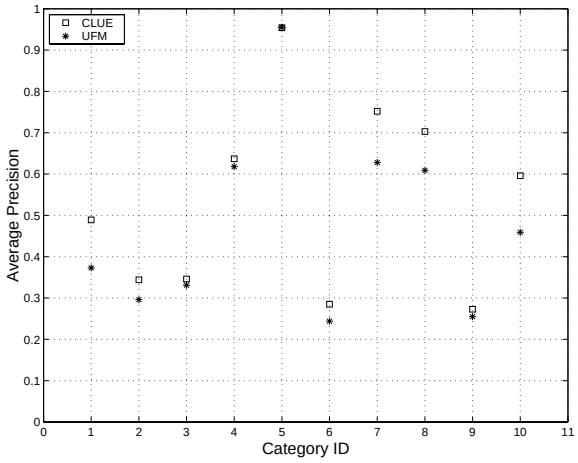

Figure 6: Comparing CLUE with UFM on the average precision.

10 categories are 0.538 for CLUE and 0.477 for UFM.

3.4 Speed

CLUE has been implemented on a Pentium III 700MHz PC running Linux operation system. To compare the speed of CLUE with UFM [4], which is implemented and tested on the same computer, 100 random queries are issued to the demonstration web sites. CLUE takes on average 0.8 second per query for similarity measure evaluation, sorting, and clustering, while UFM takes 0.7 second to evaluate similarities and sort the results. The size of the database is $60,000$ for both tests. Although CLUE is slower than UFM because of the extra computational cost for NNM and recursive Ncut, the execution time is still well within the tolerance of real-time image retrieval.

4. CONCLUSIONS AND FUTURE WORK

This paper introduces CLUE, a novel image retrieval scheme, based on a rather simple assumption: semantically similar images tend to be clustered in some feature space. CLUE at-

tempts to retrieve semantically coherent image clusters from unsupervised learning of how images of the same semantics are alike. The empirical results suggest that this assumption seems to be reasonable when target images close to the query image are under consideration. CLUE is a general approach in the sense that it can be combined with any real-valued symmetric image similarity measure (metric or non-metric). Thus it may be embedded in many current CBIR systems. The application of CLUE to a database of 60,000 general-purpose images demonstrates that CLUE can provide semantically more meaningful results to a system user than an existing CBIR system using the same similarity measure. Numerical evaluations show good cluster quality and improved retrieval accuracy.

CLUE has several limitations.

- The current heuristic used in the recursive Ncut always bipartitions the largest cluster. This is a low-complexity rule. But it may divide a large and pure cluster into several clusters even when there exists a smaller and semantically more diverse cluster.
- The current method of finding a representative image for a cluster does not always give a semantically accurate result. For the example in Figure 4(a), one would expect the representative image to be a bird image. But the system picks an image of sheep.
- If the number of neighboring target images is large, sparsity of the affinity matrix becomes crucial to retrieval speed. The current weighting scheme does not lead to a sparse affinity matrix.

One possible future direction is to integrate CLUE with keyword-based image retrieval approaches. Other graph theoretic clustering techniques need to be tested for possible performance improvement. CLUE may be combined with nonlinear dimensionality reduction techniques. CLUE may also be useful for image understanding. As future work, we intend to apply CLUE to search, browse, and learn concepts from digital imagery for Asian art and cultural heritages.

Acknowledgments

The material is based upon work supported by the National Science Foundation under Grant No. IIS-0219272, The Pennsylvania State University, the PNC Foundation, SUN Microsystems under Grant EDUD-7824-010456-US. Part of the work was completed when Y. Chen was supported by a summer internship at the NEC Research Institute. The authors would like to thank Jia Li, C. Lee Giles, Donald Richards, and John Yen for helpful discussions.

5. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval,* Addison-Wesley, 1999.

[2] K. Barnard and D. Forsyth, "Learning the Semantics of Words and Pictures," *Proc. 8th Int'l Conf. on Computer Vision*, vol. 2, pp. 408–415, 2001.

[3] C. Carson, S. Belongie, H. Greenspan, and J. Malik, "Blobworld: Image Segmentation Using Expectation-Maximization and its Application to Image Querying," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 8, pp. 1026–1038, 2002.

[4] Y. Chen and J. Z. Wang, "A Region-Based Fuzzy Feature Matching Approach to Content-Based Image Retrieval," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 9, pp. 1252–1267, 2002.

[5] I. J. Cox, M. L. Miller, T. P. Minka, T. V. Papathomas, and P. N. Yianilos, "The Bayesian Image Retrieval System, PicHunter: Theory, Implementation, and Psychophysical Experiments," *IEEE Trans. Image Processing*, vol. 9, no. 1, pp. 20–37, 2000.

[6] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, "Efficient and Effective Querying by Image Content," *J. Intell. Inform. Syst.*, vol. 3, no. 3-4, pp. 231–262, 1994.

[7] G. H. Golub and C. F. Van Loan, *Matrix Computations,* 3rd ed., Johns Hopkins University Press, 1996.

[8] Y. Gdalyahu, D. Weinshall, and M. Werman, "Self-Organization in Vision: Stochastic Clustering for Image Segmentation, Perceptual Grouping, and Image Database Organization," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 23, no. 10, pp. 1053–1074, 2001.

[9] A. Gupta and R. Jain, "Visual Information Retrieval," *Commun. ACM*, vol. 40, no. 5, pp. 70–79, 1997.

[10] M. A. Hearst and J. O. Pedersen, "Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results," *Proc. of the 19th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 76–84, 1996.

[11] D. W. Jacobs, D. Weinshall, and Y. Gdalyahu, "Classification with Nonmetric Distances: Image Retrieval and Class Representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 6, pp. 583–600, 2000.

[12] J. Li and J. Z. Wang, "Automatic Linguistic Indexing of Pictures By a Statistical Modeling Approach," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no.10, 2003.

[13] J. Li, J. Z. Wang, and G. Wiederhold, "IRM: Integrated Region Matching for Image Retrieval," *Proc. 8th ACM Int'l Conf. on Multimedia*, pp. 147–156, 2000.

[14] W. Y. Ma and B. Manjunath, "NeTra: A Toolbox for Navigating Large Image Databases," *Proc. IEEE Int'l Conf. Image Processing*, pp. 568–571, 1997.

[15] S. Mehrotra, Y. Rui, M. Ortega-Binderberger, and T. S. Huang, "Supporting Content-Based Queries over Images in MARS," *Proc. IEEE Int'l Conf. on Multimedia Computing and Systems*, pp. 632–633, June 1997.

[16] A. Pentland, R. W. Picard, and S. Sclaroff, "Photobook: Content-Based Manipulation for Image Databases," *Int'l J. Comput. Vis.*, vol. 18, no. 3, pp. 233–254, 1996.

[17] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra, "Relevance Feedback: A Power Tool for Interactive Content-Based Image Retrieval," *IEEE Trans. Circuits and Video Technology*, vol. 8, no. 5, pp. 644–655, 1998.

[18] G. Sheikholeslami, W. Chang, and A. Zhang, "SemQuery: Semantic Clustering and Querying on Heterogeneous Features for Visual Data," *IEEE Trans. Knowledge and Data Engineering*, vol. 14, no. 5, pp. 988–1002, 2002.

[19] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[20] J. R. Smith and S.-F. Chang, "VisualSEEK: A Fully Automated Content-Based Query System," *Proc. 4th ACM Int'l Conf. on Multimedia*, pp. 87–98, 1996.

[21] A. Vailaya, M. A. T. Figueiredo, A. K. Jain, and H.-J. Zhang, "Image Classification for Content-Based Indexing," *IEEE Trans. Image Processing*, vol. 10, no. 1, pp. 117–130, 2001.

[22] J. Z. Wang, J. Li, and G. Wiederhold, "SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture LIbraries," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 23, no. 9, pp. 947–963, 2001.

[23] S. C. Zhu and A. Yuille, "Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, no. 9, pp. 884–900, 1996.