CLUE: Cluster-Based Retrieval of Images by Unsupervised Learning

Yixin Chen, *Member, IEEE*, James Z. Wang, *Member, IEEE*, and Robert Krovetz

***Abstract*—In a typical content-based image retrieval (CBIR) system, target images (images in the database) are sorted by feature similarities with respect to the query. Similarities among target images are usually ignored. This paper introduces a new technique, cluster-based retrieval of images by unsupervised learning (CLUE), for improving user interaction with image retrieval systems by fully exploiting the similarity information. CLUE retrieves image clusters by applying a graph-theoretic clustering algorithm to a collection of images in the vicinity of the query. Clustering in CLUE is dynamic. In particular, clusters formed depend on which images are retrieved in response to the query. CLUE can be combined with any real-valued symmetric similarity measure (metric or nonmetric). Thus, it may be embedded in many current CBIR systems, including relevance feedback systems. The performance of an experimental image retrieval system using CLUE is evaluated on a database of around 60,000 images from COREL. Empirical results demonstrate improved performance compared with a CBIR system using the same image similarity measure. In addition, results on images returned by Google's Image Search reveal the potential of applying CLUE to real-world image data and integrating CLUE as a part of the interface for keyword-based image retrieval systems.**

***Index Terms*—Content-based image retrieval (CBIR), image classification, similarity measure, spectral graph clustering, unsupervised learning.**

I. Introduction

CONTENT-BASED image retrieval (CBIR) aims at developing techniques that support effective searching and browsing of large image digital libraries based on automatically derived imagery features. It is a rapidly expanding research area situated at the intersection of databases, information retrieval, and computer vision. 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. Readers are referred to [33] for additional references.

Manuscript received November 21, 2003; revised July 23, 2004. This work was supported in part by the National Science Foundation under Grant IIS-0219272 and CNS-0202007, The Pennsylvania State University, the PNC Foundation, SUN Microsystems under Grant EDUD-7824-010456-US, the University of New Orleans, The Research Institute for Children, NASA/EPSCoR DART under Grant NCC5-573, and the NEC Research Institute. This research was done when R. Krovetz was with NEC Research Institute, Princeton, NJ. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Gopal Pingali.

Y. Chen is with the Department of Computer Science, University of New Orleans, New Orleans, LA 70148 USA, and also with The Research Institute for Children, New Orleans, LA 70118 USA (e-mail: yixin@cs.uno.edu).

J. Z. Wang is with the School of Information Sciences and Technology and the Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA 16802 USA (e-mail: jwang@ist.psu.edu).

R. Krovetz is with Teoma Technologies, Piscataway, NJ 08554 USA (e-mail: krovetz@ask.com).

Digital Object Identifier 10.1109/TIP.2005.849770

A. Previous Work

From a computational perspective, 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 image in proportion to a similarity measure calculated from the features. In this sense, these features, or signatures of images, characterize the *content* of images. According to the scope of representation, features roughly fall into two categories: global features and local features. The former category includes texture histogram, color histogram, color layout of the whole image, and features selected from multidimensional discriminant analysis of a collection of images [8], [11], [34], [36]. While color, texture, and shape features for subimages [24], segmented regions [3], [4], [21], [40], or interest points [30] belong to the latter category.

As a key issue in CBIR, similarity measure quantifies the resemblance in contents between a pair of images [28]. Depending on the type of features, the formulation of the similarity measure varies greatly. The Mahalanobis distance [12] and intersection distance [35] are commonly used to compute the difference between two histograms with the same number of bins. When the number of bins are different, the Earth mover's distance (EMD) [26] applies. The EMD is computed by solving a linear programming problem. Moments [18], the Hausdorff metric [14], elastic matching [2], and decision trees [16] have been proposed for shape comparison. In [23], a similarity measure is defined from subjective experiments and multidimensional scaling (MDS) based upon the model of human perception of color patterns. Barnard *et al.* [1] presented a probability-based similarity measure that combines the information provided by text and the visual information provided by image features. Li *et al.* [20] presented an integrated region matching scheme for region-based image retrieval. Recently, a similarity measure using fuzzified region features is introduced in [4]. It is shown to be robust to segmentation-related uncertainties.

In one way or another, the aforementioned similarity measures capture certain facets of image content, named the *similarity-induced semantics*. Nonetheless, the meaning of an image is rarely self evident. Similarity-induced semantics usually does not coincide with the high-level concept conveyed by an image (*semantics* of the image). This is referred to as the *semantic gap* [33], which reflects the discrepancy between the relatively limited descriptive power of low-level visual features and high-level concepts. Many approaches have been proposed to reduce the semantic gap. They generally fall into two classes depending on the degree of user involvement in the retrieval: relevance feedback and image database preprocessing using statistical classification.

Relevance feedback is a powerful technique originally used in the traditional text-based information retrieval systems. In CBIR, 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 [6], [27], [42]. Based on user feedback, the model of similarity measure is dynamically updated to give a better approximation of the perception subjectivity. There are also works that combine relevance feedback with supervised learning [38]: Binary classifiers are trained on the fly based on user feedback. 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 nonrelevant).

Statistical classification methods group images into semantically meaningful categories using low-level visual features so that semantically adaptive searching methods applicable to each category can be applied [19], [31], [39], [40]. For example, the SemQuery system [31] categorizes images into different set of clusters based on their heterogeneous features. Vailaya *et al.* [39] 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. SIMPLIcity system [40] classifies images into graph, textured photograph, or nontextured photograph and, thus, narrows down the searching space in a database. ALIP system [19] uses categorized images to train hundreds of statistical models each corresponding to a semantic category. Although these classification methods are successful in their specific domains of application, the simple ontologies built upon them could not incorporate the rich semantics of a sizable image database.

B. Our Approach

All current CBIR techniques assume certain mutual information between the similarity measure and the semantics of the images. A typical CBIR system ranks target images according to the similarities with respect to the query and neglects the similarities between target images. Can we improve the performance of a CBIR system by including the similarity information between target images? This is the question we attempt to address in this work. We propose a new technique for improving user interaction with image retrieval systems by fully exploiting the similarity information. The technique, which is named cluster-based retrieval of images by unsupervised learning (CLUE), *retrieves 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 for the users semantic relevant *clues* as to where to navigate.

CLUE has the following characteristics.

- 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. Moreover, as shown in Section V-D, it may also be used as a part of the interface for keyword-based image retrieval systems.
- It uses a graph-theoretic algorithm to generate clusters. In particular, a set of images is represented as a weighted undirected graph: nodes correspond to images; an edge connects two nodes; and the weight on an edge is related to the similarity between the two nodes (or images). Graph-based representation and clustering sidestep the restriction of a metric space. This is crucial for nonmetric image similarity measures (many commonly used similarity measures are indeed nonmetric [15]).
- The clustering is local and dynamic. In this sense, CLUE is similar to the scatter/gather method proposed for document (or text) retrieval [13]. 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. This is in contrast to current image database statistical classification methods [31], [39], [40], in which the image categories are derived for the whole database in a preprocessing stage and, therefore, are global, static, and independent of the query.

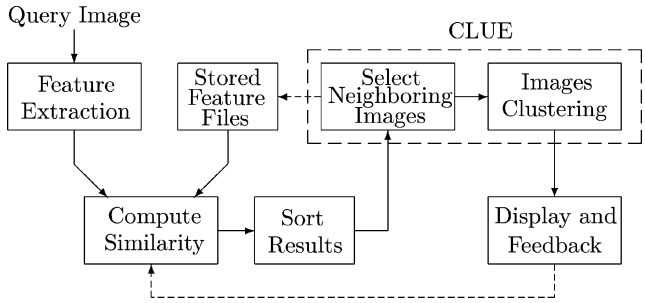


Fig. 1. Diagram of a cluster-based image retrieval system. The arrows with dotted lines may not exist for some systems.

C. Outline of the Paper

The remainder of the paper is organized as follows. Section II describes the general methodology of CLUE. A summary of the algorithm and computational issues are discussed in Section III. An image retrieval system using CLUE is introduced in Section IV. Section V presents the experimental results. Finally, we conclude in Section VI, together with a discussion of future work.

II. Retrieval of Similarity-Induced Image Clusters

In this section, we first present an overview of a cluster-based image retrieval system. We then describe in detail the major components of CLUE, namely, neighboring image selection and image clustering.

A. System Overview

From a data-flow viewpoint, a cluster-based image retrieval system can be characterized by the diagram in Fig. 1. 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 model of similarity measure according to user feedback (if relevance feedback is included).

The major difference between a cluster-based image retrieval system 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. Fig. 1 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. The only requirement is a real-valued similarity measure satisfying the symmetry property. 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. An introduction of a specific cluster-based image retrieval system, which we have implemented, will be given in Section IV.

B. Neighboring Target Images Selection

To mathematically define the neighborhood of a point, we need to first choose a measure of distance. As for 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 NN of i as seeds. The r NN 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 NN, i.e., distinct target images in $k(r+1)$ target images. Thus, the number of neighboring target images is bounded above by $k(r+1)$.

If the distance is metric, both methods would generate similar results under proper parameters (ϵ, k, and r). However, for nonmetric 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 high even if both of them are very close to a query image. Compared with the FRM, our empirical results show that, with proper choices of k and r, NNM tends to generate more structured collection of target images under a nonmetric distance. In this work, we use NNM because the image similarity measure of our experimental retrieval system is not metric. A detailed discussion of computational issues (including parameters selection) will be covered in Section III.

C. Weighted Graph Representation of a Collection of Images

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 [15]. One is called the *geometric representation*, in which data items are mapped to some real normed vector space. The other is referred to as the *graph representation* emphasizing the pairwise relationship. 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 because many distances defined for images are nonmetric for reasons given in [15]. 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 nonnegative 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

$$w_{ij} = e^{-\frac{d(i,j)^2}{s^2}} \tag{1}$$

where s is a scaling parameter that needs to be tuned to get a suitable locality. The choice of exponential decay is based on support from psychological studies provided by [10]. The weights can be organized into a matrix $\mathbf{W}$, named the *affinity matrix*, with the ijth entry given by w_{ij}. Although (1) is a relatively simple weighting scheme, our experimental results (Section V) have shown its effectiveness. The same weighting scheme has been used in [10], [32], [41].

D. Spectral Graph Partitioning

Under a graph representation, clustering can be naturally formulated as a graph partitioning problem. Among many graph-theoretic algorithms, spectral graph partitioning methods [5], [29], [32], [41] have been successfully applied to many areas in computer vision including motion analysis [5], image segmentation [32], [41], and object recognition [29]. In this paper, we use one of the techniques, the normalized cut (Ncut) method [32], for image clustering. 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 generating balanced clusters [32], [41]. Next, we present a brief review of the Ncut method based on Shi and Malik's work [32]. More exhaustive treatments can be found in [32] and [41].

Roughly speaking, a graph partitioning method attempts to organize nodes into groups so that the within-group similarity is

high, and/or the between-groups similarity is low. Given a graph $G = (\mathbf{V}, \mathbf{E})$ with affinity matrix $\mathbf{W}$, a simple way to quantify the cost for partitioning nodes into two disjoint sets $\mathbf{A}$ and $\mathbf{B}$ ($\mathbf{A} \cap \mathbf{B} = \emptyset$ and $\mathbf{A} \cup \mathbf{B} = \mathbf{V}$) is the total weights of the edges that connecting the two sets. In graph theory, this cost is called a *cut*

$$\text{cut}(\mathbf{A}, \mathbf{B}) = \sum_{i \in \mathbf{A}, j \in \mathbf{B}} w_{ij} \tag{2}$$

which can also be viewed as a measure of the between-groups similarity.

Finding a bipartition of the graph that minimizes this cut value is known as the *minimum cut* problem. There exist efficient algorithms for solving this problem. However, the minimum cut criterion favors grouping small sets of isolated nodes in the graph [32] because the cut defined in (2) does not contain any within-group information. In other words, the minimum cut usually yields over-clustered results when it is recursively applied. This motivates several modified graph partition criteria including the Ncut

$$\text{Ncut}(\mathbf{A}, \mathbf{B}) = \frac{\text{cut}(\mathbf{A}, \mathbf{B})}{\text{cut}(\mathbf{A}, \mathbf{V})} + \frac{\text{cut}(\mathbf{A}, \mathbf{B})}{\text{cut}(\mathbf{B}, \mathbf{V})}.$$

An unbalanced cut would generate a large Ncut value.

Finding a bipartition with minimum Ncut value is an NP-complete problem. Shi and Malik proposed an approximated solution by solving a generalized eigenvalue problem [32]

$$(\mathbf{D} - \mathbf{W})\vec{y} = \lambda \mathbf{D}\vec{y} \tag{3}$$

where $\mathbf{W}$ is an $n \times n$ affinity matrix, $\mathbf{D} = \text{diag}[s_1, s_2, \cdots, s_n]$ is a diagonal matrix with $s_i = \sum_{j=1,\cdots,n} w_{ij}$. Generalized eigenvector corresponding to the second smallest generalized eigenvalue (or in short the second smallest generalized eigenvector) is then used to partition the graph.

The Ncut method can be recursively applied to get more than two clusters, but this leads to the following questions: 1) Which subgraph should be divided? 2) When should the process stop? In this paper, we use a simple heuristic. The subgraph with the maximum number of nodes is recursively partitioned (random selection is used for tie breaking). The process terminates when the bound on the number of clusters is reached or the Ncut value exceeds some threshold.

E. Finding a Representative Image for a Cluster

Ultimately, the system needs to present the clustered target images to the user. Unlike a typical CBIR system, which displays certain numbers of top matched target images to the user, a cluster-based image retrieval 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 do we organize these clusters? 2) How do we find a representative image for each cluster? The organization of clusters will be described in Section III-B. 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 be $\{\mathbf{C}_1, \mathbf{C}_2, \cdots, \mathbf{C}_m\}$, which is also a partition of $\mathbf{V}$, i.e., $\mathbf{C}_i \cap \mathbf{C}_j = \emptyset$ for $i \neq j$ and $\bigcup_{i=1}^{m} \mathbf{C}_i = \mathbf{V}$. Then, the representative node (image) of $\mathbf{C}_i$ is

$$\arg\max_{j \in \mathbf{C}_i} \sum_{t \in \mathbf{C}_i} w_{jt}. \tag{4}$$

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

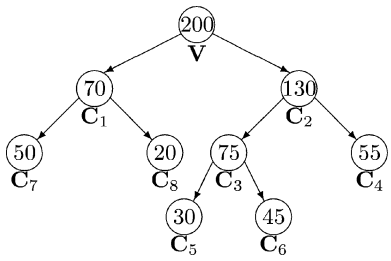


Fig. 2. Tree generated by four Ncuts that are applied to $\mathbf{V}$ with 200 nodes. The numbers denote the size of the corresponding clusters.

III. Algorithmic View

This section starts with an algorithmic summary of CLUE. We then introduce the organization of clusters, followed by a discussion of computational complexity and parameters selection.

A. Outline of the Algorithm

The inputs for CLUE include a query image, $k \geq 2$ and $r \geq 1$ needed by NNM for neighboring target images selection, maximum number of clusters ($M \geq 2$), and threshold for the Ncut value ($0 \leq T \leq 1$) required by the recursive Ncut process. CLUE first selects a collection of neighboring target images for a query image using NNM. Next, it constructs a weighted undirected graph containing the query image and its neighboring target images. It then applies the Ncut algorithm recursively to the graph or the largest subgraph until the number of clusters is equal to M or the Ncut value is greater than T. Finally, the representative images for the clusters are found according to (4).

B. Organization of Clusters

The recursive Ncut partition is essentially a hierarchical divisive clustering process that produces a tree. For example, Fig. 2 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. In data visualization, displaying a tree is a commonly used technique. However, the tree organization here may not be

useful to a user because there is no guarantee of any correspondence between the tree and the semantic structure of images. For example, Fig. 6 shows clusters and the associated tree generated by CLUE for a query image of food. Note that food and horses are leaf nodes of the same parent, but the semantic meaning of their parent node is not clear. Thus, it may be extremely difficult to find representative images for parent nodes. Moreover, those representative images of parent nodes may be misleading to a user. 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). For the example in Fig. 2, it yields a sequence: $\mathbf{C}_7$, $\mathbf{C}_8$, $\mathbf{C}_5$, $\mathbf{C}_6$, and $\mathbf{C}_4$. However, the order of two clusters produced by an Ncut bipartition iteration is still undecided, i.e., which one should be the *left child* and which one should be the *right child*. This can be solved by enforcing 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.

The traversal ordering and arbitration rule have the following properties.

- The query image is in the leftmost leaf ($\mathbf{C}_7$ in Fig. 2) 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 (note that $\mathbf{V}$ includes the query image).
- We can view d_1 (or d_2) as a distance from a query image to a cluster of images. In this sense, for any parent node, its left child is closer to the query image than its right child.
- In the traversal, the leaves of the left subtree of any parent node appear before the leaves of its right subtree.

Therefore, the resulting linear organization of clusters considers not only the distances to a query image, but also the hierarchical structure that generates the clusters. To this end, it may be viewed as a structured sorting of clusters in ascending order of distances to a query image. For the sake of consistency, images within each cluster are also organized in ascending order of distances to the query.

C. Computational Complexity

The computational complexity of a cluster-based image retrieval system is higher than that of a typical CBIR system due to the added computation of clustering. The time complexity of CLUE is the sum of the complexity of NNM and the complexity of the recursive Ncut.

Since NNM needs to find r NN for all k seeds, a straightforward implementation, which treats each seed as a new query, would make the whole process very slow when the size of image database is large. Two methods can be applied to reduce the time cost of NNM. One method is to parallelize NNM because NN for all k seeds can be selected simultaneously. The other method utilizes the fact that all seeds are images in the database. Thus, similarities can be computed and sorted in advance. So the time needed by NNM does not scale up by the number of seeds. Nevertheless, it then requires storing the sorting results with every image in the database as a query image. The space complexity becomes $O(N^2)$ where N is the size of the database. However, the space complexity can also be reduced because NNM only needs r NN, which leads to a space complexity of $O(rN)$. The locality constraint guarantees that r is very small compared with N. In our implementation, only the ID numbers of 100 NN for each image are stored ($N = 60\,000$). The second method is used in our experimental system. We argue that this method is practical even if the database is very large. Because computing and sorting similarities for all target images may be very time-consuming, this process is required only once. Moreover, the process can also be parallelized for each target image. If new images are added to the database, instead of redoing the whole process, we can merely compute those similarities associated with new images and update previously stored sorting results accordingly.

The time needed by the recursive Ncut process consists of two parts: graph construction and the Ncut algorithm. For graph construction, one needs to evaluate $n(n+1)/2$ entries of the affinity matrix where $n \leq k(r+1)+1$ is the number of nodes (query image and all its neighboring target images). Thus, the time complexity is $O(n^2)$. Each Ncut iteration can be solved by the Lanczos algorithm [9, ch. 9] in $O(n^2)$ [32]. As the number of clusters is bounded by M, the total time complexity for the recursive Ncut process is $O(k^2r^2)$ (because $n \leq k(r+1)+1$).

D. Parameters Selection

Several parameters need to be specified to implement CLUE. These include k and r for NNM, s for affinity matrix evaluation, and M and T for recursive Ncut. Three requirements are considered when deciding k and r. First, we want the neighboring images to be close to the query image so that the assumption of a locally clustered structure is valid. Second, we need sufficient number of images to provide an informative local visualization of the image database to the user. Third, computational cost should be kept within the tolerance of real-time applications. It is clear that the second constraint favors large k and r, while the other two constraints need k and r to be small. Finding a proper tradeoff is application dependent.

For the cluster-based image retrieval system described in the next section, k and r are obtained from a simple tuning strategy. We randomly pick 20 query images from the image database. For each pair of k and r, where $k \in \{25, 26, \cdots, 35\}$ and $r \in \{5, 6, \cdots, 10\}$, we manually examine the semantics of images generated by NNM using each of the 20 query images, and record the average number of distinct semantics. Next, all pairs of k and r corresponding to the median of the above recorded numbers are found. We pick the pair with minimal kr value, which gives $k = 29$ and $r = 7$ for our system. As a byproduct, M (maximum number of clusters) in recursive Ncut is set to be 8, which is the integer closest to the median. Note that our criteria on distinct semantics may be very different from the criteria of a system user. However, we observed that the system is not sensitive to k and r.

The parameter s in (1) reflects the local scale on distances. Thus, it should be adaptive to the query image and its neighboring target images. In our system, $s = 2\sigma$ where σ is the standard deviation of all the pairwise distances used to construct the affinity matrix. The threshold T is chosen to make the median of the number of clusters generated by recursive Ncuts on

(a)

(b)

Fig. 3. Two snapshots of the user interface displaying query results for a query image with ID 6275. (a) List of image cluster thumbnails. (b) Images in Cluster 1. Below each thumbnail in (a) are cluster ID and the number of images in the cluster. There are two numbers below images in (b). The underlined number is image ID. The other number is the cluster ID.

the 20 collections of images, which are used in k and r tuning process, equal or close to $M = 8$. A proper T value is found to be 0.9.

IV. Content-Based Image Clusters Retrieval System

The system uses the same feature extraction scheme and similarity measure as those in [4] and [40]. So only a brief introduction is presented here. From the viewpoint of image features, the system is a region-based system. It applies image segmentation to decompose an image into regions, and defines similarities via region matching. To segment an image, our system first partitions the image into blocks with 4×4 pixels. A feature vector, consisting of six features, is then extracted for each image block. Among the six features, three of them are the average color (in the LUV color space) of the corresponding block. The other three represent energy in the high-frequency bands of a one-level Daubechies-4 wavelet transform [7] applied to the L component of the image block, that is, the square root of the second-order moment of wavelet coefficients in the LH (low high), HL, and HH bands. The k-means algorithm is then used to group the feature vectors into several classes with every class corresponding to one region in the segmented image.

Each region is then associated with a fuzzy feature (defined by a *membership function*) describing the color, texture, and shape properties of the region. The membership functions of fuzzy sets naturally characterize the gradual transition between regions within an image. To that end, they characterize the blurring boundaries due to imprecise segmentation. A fuzzy similarity measure is used to describe the resemblance of two regions. Finally, a convex combination scheme synthesizes the region-level similarities into an image similarity measure, unified feature matching (UFM) measure, which is demonstrated to be very robust to segmentation-related uncertainties [4]. In order to compute the affinity matrix according to (1), UFM measure is converted to a distance by a simple linear transformation $d(i, j) = 1 - \mathrm{UFM}(i, j)$.

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. The system is capable of handling any standard image format from anywhere on the Internet and reachable by our server via the HTTP protocol. 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 (4), and sorted using the algorithm in Section III-B. Fig. 3(a) shows eight clusters corresponding to a query image with ID 6275. Below each thumbnail are cluster ID and the number of images in that cluster. A user can start a new query search by submitting a new image ID or URL, get a random set of images from the image database, or click a thumbnail to see all images in the associated cluster. The contents of Cluster 1 are displayed in Fig. 3(b). From left to right and top to bottom, the images are listed in ascending order of distances to the query image. The underlined numbers below the images are image IDs. The other numbers are cluster IDs. The image with a border around it is the representative image for the cluster. Again, a user has

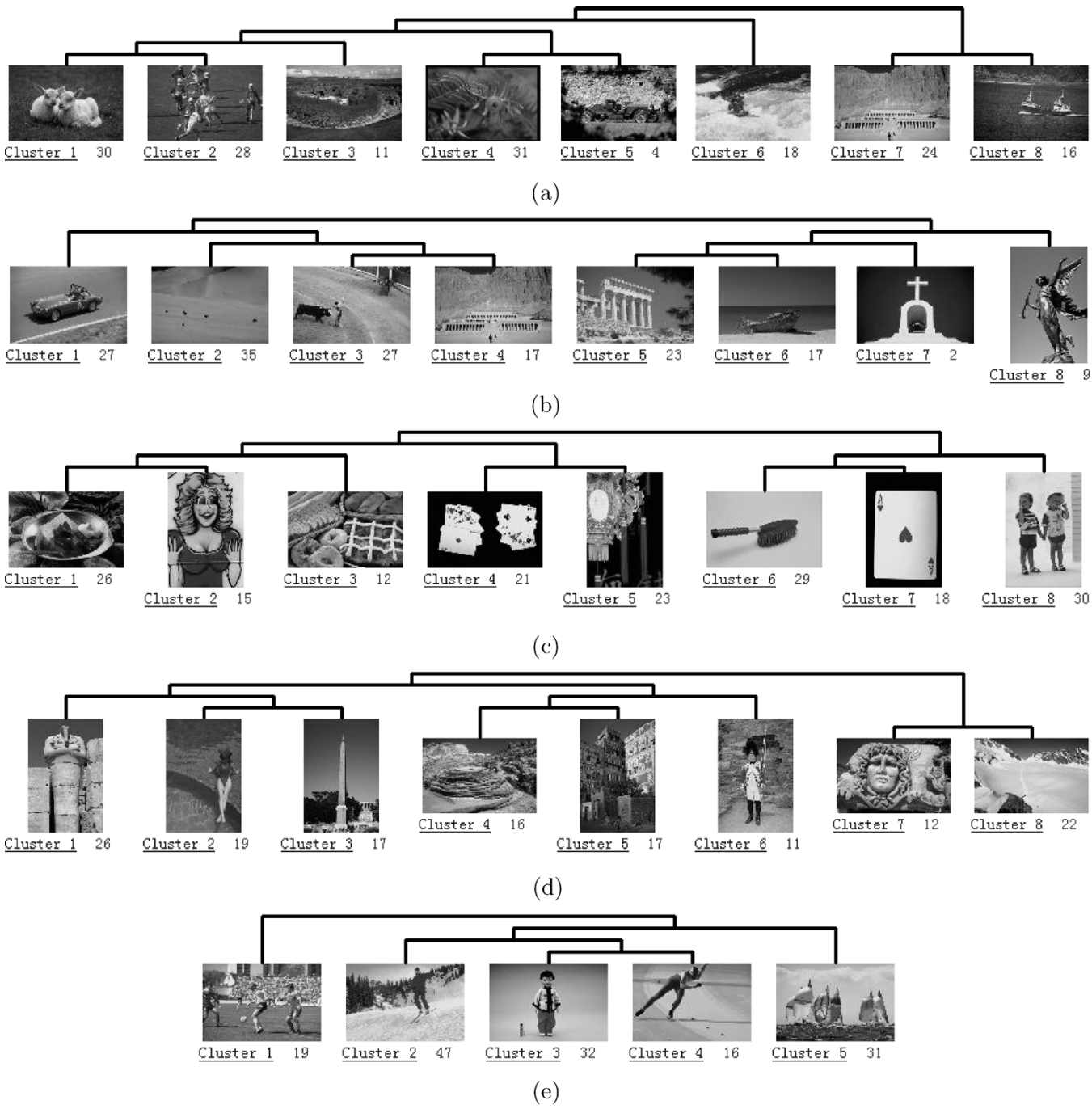


Fig. 4. Image clusters and the associated tree structure generated by CLUE. (a) Birds. (b) Car. (c) Food. (d) Historical building. (e) Soccer game.

three options: enter a new image ID or URL, get a random set of images from the database, or click an image to submit it as a query.

V. Experiments

Our system[1] is implemented with a general-purpose image database (from COREL), which includes about 60 000 images stored in JPEG format with size 384×256 or 256×384. In Section V-A, we provide several query results on the COREL database to intuitively illustrate the performance of the system. Section V-B presents systematic evaluations of CLUE algorithm in terms of the goodness of image clustering and retrieval accuracy. Numerical comparisons with the SIMPLIcity system using UFM similarity measure [4] are also given. In Section V-C, the speed of CLUE is compared with that of a typical CBIR system using UFM similarity measure. Section V-D presents results on images returned by Google's Image Search.

A. 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*. The image clusters and the associated tree structure generated by CLUE are shown in Fig. 4,

[1]A demonstration system is available at http://wang.ist.psu.edu/IMAGE/clue.

Fig. 5. 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 building. (e) Soccer game.

where each image thumbnail represents an image cluster. Below each thumbnail are cluster ID and the number of images in the cluster. 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 III-B. Performance issues about the rest clusters will be covered in Section V-B. Since CLUE of our system 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 Fig. 5, 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 Fig. 5. 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 all query examples given in Fig. 5. 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-cluster similarity is low. The results seem to indicate that a similarity-induced image cluster tends to contain images of similar semantics. In other words, organizing images into clusters and retrieving image clusters may help to reduce the semantic gap even when the rest of the components of the system, such as feature extraction and image similarity measure, remain unchanged.

B. Systematic Evaluation

To provide a more objective evaluation and comparison, CLUE (built upon UFM similarity measure) is tested on a subset of the COREL database, formed by ten image categories, each containing 100 images. The categories are Africa people and villages, beach, buildings, buses, dinosaurs, elephants, flowers, horses, mountains and glaciers, 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 category (or 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 ten categories were chosen so that each depicts a distinct semantic topic.

1) Measuring the Quality of Image Clustering: Ideally, a cluster-based image retrieval system 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. Although we can force CLUE to always generate ten 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 $\mathbf{C}_j$, $j = 1, \cdots, m$. Cluster $\mathbf{C}_j$'s purity can be defined as

$$p(\mathbf{C}_j) = \frac{1}{|\mathbf{C}_j|} \max_{k=1,\cdots,c} |\mathbf{C}_{j,k}| \tag{5}$$

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 $[1/c, 1]$ 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:

$$h(\mathbf{C}_j) = -\frac{1}{\log c} \sum_{k=1}^{c} \frac{|\mathbf{C}_{j,k}|}{|\mathbf{C}_j|} \log \frac{|\mathbf{C}_{j,k}|}{|\mathbf{C}_j|}. \tag{6}$$

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. For example, if half of the images of a cluster belong to one semantic class and the rest of the images are evenly divided into nine different semantic classes, then the entropy is 0.7782 and the purity is 0.5. Fig. 6 shows clusters and the associated tree structure generated by CLUE for a sample query image of food. Size of each cluster, purity, and entropy of leaf clusters are also listed.

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., $P(i) = (1/m_i) \sum_{j=1}^{m_i} p(\mathbf{C}_j)$ where $p(\mathbf{C}_j)$ is computed according to (5); and 4) $H(i)$ is the *average entropy* of the retrieved clusters, i.e., $H(i) = (1/m_i) \sum_{j=1}^{m_i} h(\mathbf{C}_j)$, where $h(\mathbf{C}_j)$ is computed according to (6).

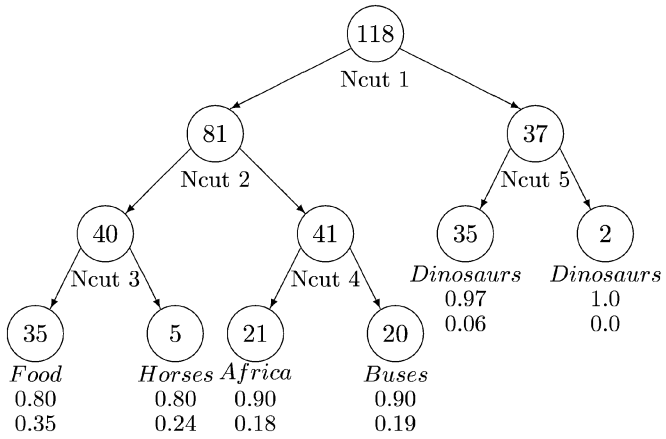


Fig. 6. CLUE applies five Ncuts to a collection of 118 images neighboring to a query image of *food*. Numbers within each node denote the size of the corresponding clusters. Linguistic descriptor and numbers listed under each leaf node are (from top to bottom): name of the dominant semantic category in the leaf node (or cluster), purity of the cluster, and entropy of the cluster.

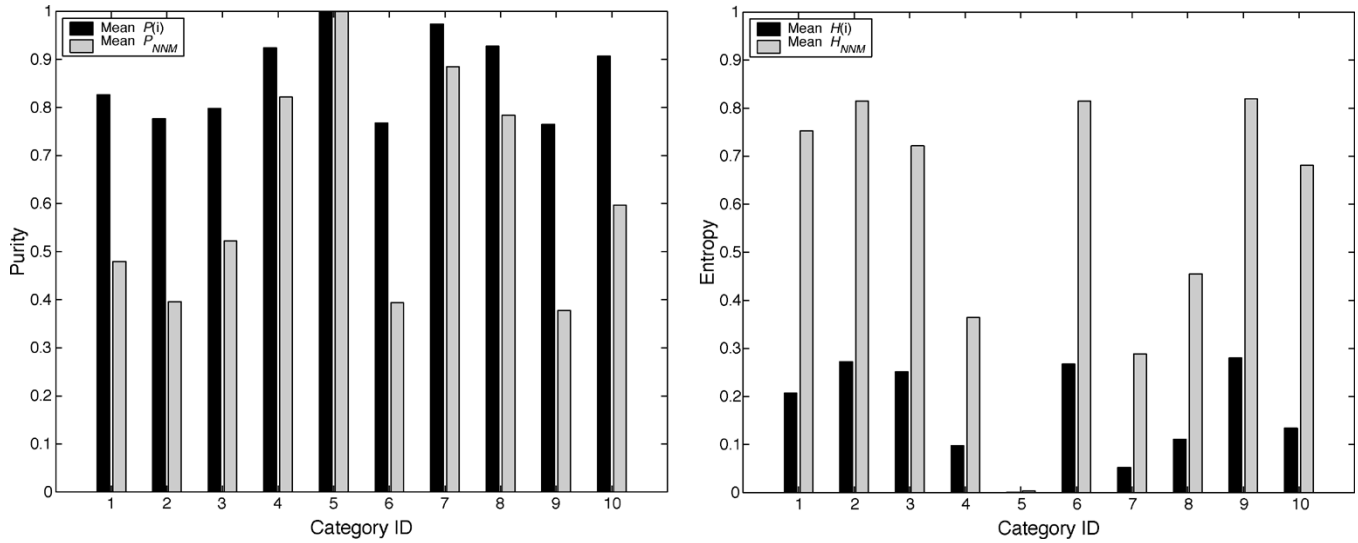


Fig. 7. Clustering performance in terms of purity and entropy. For mean $P(i)$ and mean P_{NNM}, larger numbers indicate purer clusters. For mean $H(i)$ and mean H_{NNM}, smaller numbers denote better cluster quality.

Every image in the 1000-image database is tested as a query. The same set of parameters specified in Section III-D is used here. 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 within one semantic category. The results are summarized in Table I (second and third columns) and Fig. 7. The third column of Table I 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 Fig. 7 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 image categories except category 5, for which NNM generates quite pure collections of images leaving little room for improvement.

2) Retrieval Accuracy: 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*.

A query image is correctly categorized if the dominant category in the query image cluster (first cluster of leftmost leaf) is identical to the query category. The correct categorization rate C_t for image category t indicates how likely the dominant semantics of the query image cluster coincides with the query semantics and is defined as the ratio of the number of correctly categorized images in category t to the size of category t. The fourth column of Table I lists estimations of C_t for ten categories used in our experiments. Note that randomly assigning a dominant category to the query image cluster will give a C_t value of 0.1. The results there indicate that CLUE has some difficulties in categorizing images about beaches (category 2) and images about mountains and glaciers (category 9), even though the performance is still four times better than random. A detailed examination of the errors shows that most errors on these two categories are errors between these two categories, i.e., a beach query is categorized as mountains and glaciers, or conversely. The performance degradation on these two categories seems understandable. Many images from these two categories are visually similar. Fig. 8 presents 12 images from both categories of the 1000-image database. All beach images in Fig. 8 contain mountains or mountain-like regions, while all the mountain images have regions corresponding to river, lake, or even ocean. In addition, UFM measure may also mistakenly view a glacier as clouds because both regions have similar white color and shape. However, we argue that the performance may be improved if a better similarity measure is used.

TABLE I
STATISTICS OF THE AVERAGE NUMBER OF CLUSTERS m_i AND THE AVERAGE CLUSTER SIZE v_i AND AN ESTIMATION OF THE CORRECT CATEGORIZATION RATE C_t

ID. Category Name	Mean m_i	Mean $v_i \pm$ STDV	C_t
1. Africa people and villages	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 and glaciers	7.84	14.0 ± 2.70	0.51
10. Food	7.79	12.2 ± 2.48	0.78

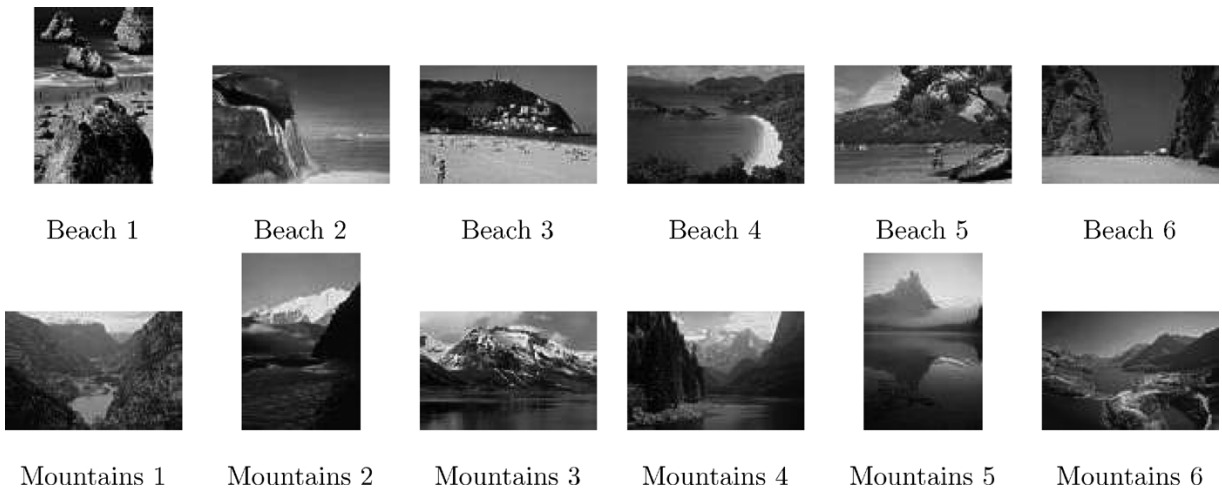


Fig. 8. Some sample images taken from the 1000-image database. They belong to two categories: beach or mountains and glaciers.

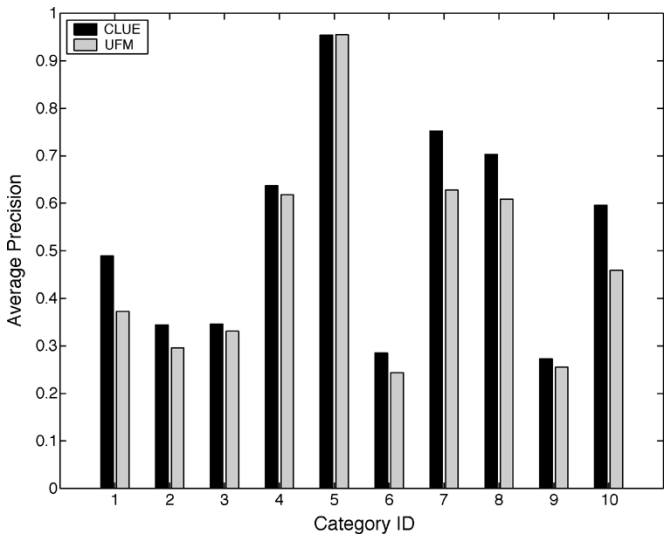


Fig. 9. Comparing CLUE scheme with UFM method on the average precision.

From the standpoint of a system user, the correct categorization rate 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 of the clusters. So we use *precision* to measure how likely a user would find images belonging to the query category within a certain number of top matches. Here, the precision is computed as the percentage of images belonging to the category of the 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 NNM is set to be 30 to ensure that the number of neighboring images generated is greater than 100. As mentioned in Section III-B, the linear organization of clusters may be viewed as a structured sorting of clusters in ascending order of distances to a query image (recall that images within each cluster are organized in ascending order of distances to the query). 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 precision for query images in category t. Fig. 9 compares the average precision given by CLUE with those obtained by UFM. Clearly, CLUE performs better than UFM for nine out of ten categories (they tie on the remaining one category). The overall average precision for ten categories are 0.538 for CLUE and 0.477 for UFM. CLUE can be built upon any real-valued symmetric similarity measure, not just UFM similarity measure. The results here suggest that on average CLUE scheme may improve the precision of a CBIR system.

C. Speed

The CLUE has been implemented on a Pentium III 700-MHz PC running the Linux operation system. To compare the speed of the CLUE with the UFM [4], which is implemented and tested on the same computer, 100 random queries are issued to the demonstration web sites. The CLUE takes on average 0.8 s per query for similarity measure evaluation, sorting, and clustering, while the UFM takes 0.7 s to evaluate similarities and sort the results. The size of the database is 60 000 for both tests. Although the CLUE is slower than the 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.

D. Application of CLUE to Web Image Retrieval

To show the performance of CLUE on real world image data, we provide some results using images from the Internet. The images are obtained from Google's Image Search (http://images.google.com), which is a keyword-based image retrieval system. Due to space limitation, we only present the results for two queries: "Tiger" and "Beijing." Since there is no query image, the neighboring image selection stage of CLUE is skipped. Instead, for each query, the recursive Ncut is directly applied to the top 200 images returned by Google. Fig. 10 lists some sample images from the top four largest clusters for each query. Each block of images are chosen to be the top 18 images within a cluster that are closest to the representative image of the cluster in terms of UFM similarity measure. The cluster size is also specified below each block of images.

As shown in Fig. 10, real-world images can be visually and semantically quite heterogeneous, even when a very specific cat-

Fig. 10. Some sample images of the top four largest clusters obtained by applying CLUE to images returned by Google's Image Search with queries "Tiger" (left column) and "Beijing" (right column). (a) Cluster 1 (75 images). (b) Cluster 2 (64 images). (c) Cluster 3 (32 images). (d) Cluster 4 (24 images). (e) Cluster 1 (61 images). (f) Cluster 2 (59 images). (g) Cluster 3 (43 images). (h) Cluster 4 (31 images).

egory is under consideration. For example, the Tiger images returned by Google's Image Search contains images of cartoon tiger (animal), real tiger (animal), Tiger Woods (golf player), tiger tank, *Crouching Tiger Hidden Dragon* (movie), tiger shark, etc. Images about Beijing include images of city maps, people, buildings, etc. CLUE seems to be capable of providing visually coherent image clusters with reduced semantic diversity within each cluster.

- The images in Fig. 10(a) are mainly about cartoon tigers. Half of the images in Fig. 10(d) contain people. Real tigers appear more frequently in Fig. 10(b) and (c) than in Fig. 10(a) and (b). Images in Fig. 10(c) have stronger textured visual effect than images of the other three blocks.
- As for images about "Beijing," the majority of the images in Fig. 10(e) are city maps. Out of the 18 images in Fig. 10(f), 11 contains people. The majority of images in Fig. 10(g) are about Beijing's historical buildings. There also a lot of images of buildings in Fig. 10(h), but most of them are modernbuilt.

These results demonstrate that, to some extent, CLUE is helpful in disambiguating and refining image semantics and, hence, improve the performance of a keyword-based image retrieval system.

VI. Conclusions and Future Work

This paper introduces CLUE, a new image retrieval scheme for improving user interaction with image retrieval systems. CLUE retrieves image clusters rather than sorted single images as most CBIR systems do. Clustering is performed in a query-dependent way. Therefore, CLUE generates clusters that are tailored to characteristics of the query image. CLUE employs a graph representation of images: Images are viewed as nodes and similarities between images are denoted by weights of the edges connecting the nodes. Clustering is then naturally formulated as a graph partitioning problem, which is solved by Ncut technique. Graph-theoretic clustering enables CLUE to handle the metric and nonmetric similarity measures in a uniform way. In this sense, CLUE is a general approach that can be combined with any real-valued symmetric image similarity measure and, thus, 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 better semantically relevant clues to a system user than an existing CBIR system using the same similarity measure. Numerical evaluations on a 1000-image database show good cluster quality and improved retrieval accuracy. Furthermore, results on images returned by Google's Image Search suggest the potential of applying CLUE to real world image data and integrating CLUE as a part of the interface for keyword-based image retrieval systems.

CLUE has the following limitations.

- The current heuristic used in the recursive Ncut always bipartitions the largest cluster. This is a low-complexity rule and is computationally efficient to implement, but it may divide a large and pure cluster into several clusters, even when there exists a smaller and semantically more diverse cluster. Bipartitioning the semantically most diverse cluster seems to be more reasonable, but the open question is how to automatically and efficiently estimate the semantic diversity of a cluster.
- The current method of finding a representative image for a cluster does not always give a semantically representative image. For the example in Fig. 5(a), one would expect the representative image to be a bird image, but the system picks an image of sheep (the third image). This discrepancy is due to the semantic gap: An image that is most similar to all images in the cluster in terms of a similarity measure does not necessarily belong to the dominant semantic class of the cluster.
- If the number of neighboring target images is large (more than several thousand), sparsity of the affinity matrix becomes crucial to retrieval speed. The current weighting scheme given by (1) does not lead to a sparse affinity matrix. As a result, different weighting schemes should be studied to improve the scalability of CLUE.

CLUE may be improved as follows.

- The quality of the clusters depends on the choice of the partitioning algorithm. Although experiments show that Ncut method produces robust clusters, other graph theoretic clustering techniques [22] need to be tested for possible performance improvement.
- As pointed out by a reviewer of the initial draft, the computational overhead for the NNM technique could be reduced if an indexing technique, such as SR tree [17], is used.
- CLUE relies heavily on the similarity measure employed. If a similarity measure does not capture semantic relevant information, clustering itself does not bridge the gap between low-level features and high-level concepts. Therefore, applying CLUE to the results of a semantic search engine, such as a keyword-based image retrieval system, like Google's Image Search, may be helpful in reducing the number of visually similar images a user needs to browse through. As future work, we intend to apply CLUE to search, browse, and learn concepts from digital imagery for Asian art and cultural heritages.
- CLUE may be combined with nonlinear dimensionality reduction techniques, such as the methods in [25] and [37], to provide a global visualization together with a local retrieval.

Acknowledgment

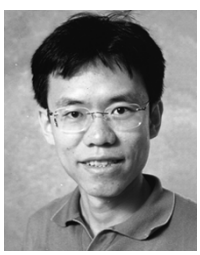
The authors would like to thank the anonymous reviewers and the Associate Editor for their comments, which led to improvements of this paper. They would also like to thank J. Li, C. L. Giles, D. Richards, and J. Yen for helpful discussions.

References

[1] K. Barnard, P. Duygulu, D. Forsyth, N. De Freitas, D. M. Blei, and M. I. Jordan, "Matching words and pictures," *J. Mach. Learn. Res.*, vol. 3, pp. 1107–1135, 2003.

[2] A. Del Bimbo and P. Pala, "Visual image retrieval by elastic matching of user sketches," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 2, pp. 121–132, Feb. 1997.

[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. Mach. Intell.*, vol. 24, no. 8, pp. 1026–1038, Aug. 2002.
[4] Y. Chen and J. Z. Wang, "A Region-Based Fuzzy Feature Matching Approach to Content-Based Image Retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 9, pp. 1252–1267, Sep. 2002.
[5] J. Costeira and T. Kanade, "A multibody factorization method for motion analysis," in *Proc. Int. Conf. Computer Vision*, 1995, pp. 1071–1076.
[6] 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 Process.*, vol. 9, no. 1, pp. 20–37, Jan. 2000.
[7] I. Daubechies, *Ten Lectures on Wavelets*. Montpelier, VT: Capital City Press, 1992.
[8] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, "Efficient and effective querying by image content," *J. Intell. Inf. Syst.*, vol. 3, no. 3–4, pp. 231–262, 1994.
[9] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: Johns Hopkins Univ. Press, 1996.
[10] Y. Gdalyahu, D. Weinshall, and M. Warman, "Self-organization in vision: stochastic clustering for image segmentation, perceptual grouping, and image database organization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 10, pp. 1053–1074, Oct. 2001.
[11] A. Gupta and R. Jain, "Visual information retrieval," *Commun. ACM*, vol. 40, no. 5, pp. 70–79, 1997.
[12] J. Hafner, H. S. Sawhney, W. Equitz, M. Flickner, and W. Niblack, "Efficient color histogram indexing for quadratic form distance functions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 17, no. 7, pp. 729–736, Jul. 1995.
[13] M. A. Hearst and J. O. Pedersen, "Reexamining the cluster hypothesis: scatter/gather on retrieval results," in *Proc. 19th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, 1996, pp. 76–84.
[14] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 9, pp. 850–863, Sep. 1993.
[15] D. W. Jacobs, D. Weinshall, and Y. Gdalyahu, "Classification with nonmetric distances: image retrieval and class representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 6, pp. 583–600, Jun. 2000.
[16] L. Jia and L. Kitchen, "Object-based image similarity computation using inductive learning of contour-segment relations," *IEEE Trans. Image Process.*, vol. 9, no. 1, pp. 80–87, Jan. 2000.
[17] N. Katayama and S. Satoh, "The SR-tree: an index structure for high-dimensional nearest neighbor queries," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1997, pp. 369–380.
[18] A. Kontanzad and Y. H. Hong, "Invariant Image Recognition by Zernike moments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 5, pp. 489–497, May 1990.
[19] J. Li and J. Z. Wang, "Automatic linguistic indexing of pictures by a statistical modeling approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 9, pp. 1075–1088, Sep. 2003.
[20] J. Li, J. Z. Wang, and G. Wiederhold, "IRM: integrated region matching for image retrieval," in *Proc. 8th ACM Int. Conf. Multimedia*, 2000, pp. 147–156.
[21] W. Y. Ma and B. Manjunath, "NeTta: a toolbox for navigating large image databases," in *Proc. IEEE Int. Conf. Image Processing*, 1997, pp. 568–571.
[22] D. W. Matula, "Graph theoretic techniques for cluster analysis algorithm," in *Classification and Clustering*, J. Van Ryzin, Ed. New York: Academic, 1977, pp. 95–129.
[23] A. Mojsilovic, J. Kovacevic, J. Hu, R. J. Safranek, and S. K. Ganapathy, "Matching and retrieval based on the vocabulary and grammar of color patterns," *IEEE Trans. Image Process.*, vol. 9, no. 1, pp. 38–54, Jan. 2000.
[24] R. W. Picard and T. P. Minka, "Vision texture for annotation," *J. Multimedia Syst.*, vol. 3, no. 1, pp. 3–14, 1995.
[25] S. T. Rowels and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, 2000.
[26] Y. Rubner, L. J. Guibas, and C. Tomasi, "The earth mover's distance, multi-dimensional scaling, and color-based image retrieval," in *Proc. DARPA Image Understanding Workshop*, May 1997, pp. 661–668.
[27] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra, "Relevance feedback: a power tool for interactive content-based image retrieval," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 5, pp. 644–655, Aug. 1998.
[28] S. Santini and R. Jain, "Similarity measures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 9, pp. 871–883, Sep. 1999.
[29] S. Sarkar and P. Soundararajan, "Supervised learning of large perceptual organization: graph spectral partitioning and learning automata," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 5, pp. 504–525, May 2000.
[30] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 5, pp. 530–535, May 1997.
[31] G. Sheikholeslami, W. Chang, and A. Zhang, "SemQuery: semantic clustering and querying on heterogeneous features for visual data," *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 5, pp. 988–1002, May 2002.
[32] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[33] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1349–1380, Dec. 2000.
[34] J. R. Smith and S.-F. Chang, "VisualSEEK: a fully automated content-based query system," in *Proc. 4th ACM Int. Conf. Multimedia*, 1996, pp. 87–98.
[35] M. J. Swain and B. H. Ballard, "Color indexing," *Int. J. Comput. Vis.*, vol. 7, no. 1, pp. 11–32, 1991.
[36] D. L. Swets and J. Weng, "Using discriminant eigenfeatures for image retrieval," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 8, pp. 831–837, Aug. 1996.
[37] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, pp. 2319–2323, 2000.
[38] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *Proc. 9th ACM Int. Conf. Multimedia*, 2001, pp. 107–118.
[39] A. Vailaya, M. A. T. Figueiredo, A. K. Jain, and H.-J. Zhang, "Image classification for content-based indexing," *IEEE Trans. Image Process.*, vol. 10, no. 1, pp. 117–130, Jan. 2001.
[40] J. Z. Wang, J. Li, and G. Wiederhold, "SIMPLIcity: semantics-sensitive integrated matching for picture libraries," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 9, pp. 947–963, Sep. 2001.
[41] Y. Weiss, "Segmentation using eigenvectors: a unifying view," in *Proc. Int. Conf. Computer Vision*, 1999, pp. 975–982.
[42] X. S. Zhou and T. S. Huang, "Relevance feedback for image retrieval: a comprehensive review," *Multimedia Syst.*, vol. 8, no. 6, pp. 536–544, 2003.



Yixin Chen (S'99–M'03) received the B.S. degree from the Department of Automation, Beijing Polytechnic University, Beijing, China, in 1995, the M.S. degree in control theory and applications from Tsinghua University, Beijing, in 1998, the M.S. and Ph.D. degrees in electrical engineering from the University of Wyoming, Laramie, in 1999 and 2001, respectively, and the Ph.D. degree in computer science from The Pennsylvania State University, University Park, in 2003.

Since August 2003, he has been an Assistant Professor with the Department of Computer Science, University of New Orleans, New Orleans, LA. His research interests include machine learning, computer vision, bioinformatics, robotics and control, and soft computing.

Dr. Chen is a member of the ACM, the IEEE Computer Society, the IEEE Neural Networks Society, and the IEEE Robotics and Automation Society.

James Z. Wang (M'96–M'00) received the B.S. degree (summa cum laude) in mathematics and computer science from the University of Minnesota, Twin Cities, in 1994, the M.Sc. degree in mathematics and the M.Sc. degree in computer science from Stanford University, Stanford, CA, in 1997, and the Ph.D. degree in medical information sciences from the Stanford University Biomedical Informatics Program and Computer Science Database Group in 2000.

Since 2000, he has been the holder of the endowed PNC Technologies Career Development Professorship and an Assistant Professor at the School of Information Sciences and Technology and the Department of Computer Science and Engineering, The Pennsylvania State University, University Park. He is a member of the DELOS-NSF Working Group on Digital Imagery for Significant Cultural and Historical Materials. He has been a visiting scholar at Uppsala University, Uppsala, Sweden, SRI International, IBM Almaden Research Center, and NEC Computer and Communications Research Laboratory.

Robert Krovetz received the B.S. degree (with honors) in computer science from the State University of New York at Stony Brook in 1977, the M.S. degree in computer science from the University of Maryland, College Park, in 1979, and the Ph.D. degree in computer science from the University of Massachusetts, Amherst, in 1995.

He was a Scientist at the NEC Research Institute, Princeton, NJ, from 1995 to 2004. Since August 2004, he has been a Senior Research Scientist at Ask Jeeves and Manager of the Natural Language Group at Teoma Technologies, Piscataway, NJ.

Dr. Krovetz is a member of the ACM, the American Society for Information Science and Technology, and the Association for Computational Linguistics.