Graph–Theoretic Clustering for Image Grouping and Retrieval

Selim Aksoy and Robert M. Haralick
Intelligent Systems Laboratory
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500
{*aksoy,haralick*}*@isl.ee.washington.edu*

Abstract

Image retrieval algorithms are generally based on the assumption that visually similar images are located close to each other in the feature space. Since the feature vectors usually exist in a very high dimensional space, a parametric characterization of their distribution is impossible, so non-parametric approaches, like the k-nearest neighbor search, are used for retrieval.

This paper introduces a graph–theoretic approach for image retrieval by formulating the database search as a graph clustering problem by using a constraint that retrieved images should be consistent with each other (close in the feature space) as well as being individually similar (close) to the query image. The experiments that compare retrieval precision with and without clustering showed an average precision of 0.76 after clustering, which is an improvement by 5.56% over the average precision before clustering.

1. Motivation

Computing feature vectors is an essential step in image database retrieval algorithms like in many computer vision and pattern recognition applications. Usually these feature vectors exist in a very high dimensional space where a parametric characterization of the distribution is often impossible. Due to the high dimensionality, this problem is usually not studied and non-parametric approaches, like the k-nearest neighbor search, are used for retrieval.

In an image database retrieval application, we expect to have visually similar images close to each other in the feature space. Unfortunately, none of the existing feature extraction algorithms can always map visually similar images to nearby locations. A common observation in retrieval results is that sometimes images that are quite irrelevant to the query image are also retrieved simply because they are close to the query image. We believe that an efficient retrieval algorithm should be able to retrieve images that are not only close (similar) to the query image but also close (similar) to each other.

In order to understand the behavior of the features, which will help us determine the effectiveness of both the features and the distance measures in establishing similarity between images, clustering the feature space and visually examining the consistency of the results is important. In their Blobworld system, Carson *et al.* [3] used the expectation-maximization algorithm to cluster the blob space to find representative blobs that can mimic human queries. They noted that their clustering procedure tends to ignore the blobs which have the best chance of distinguishing among categories because the most distinctive blobs in a given category occur much less often than less distinctive blobs.

In this paper we introduce a graph–theoretic approach for image retrieval by formulating the database search as a graph clustering problem. We also use the idea in Carson *et al.* [3] that clusters contain visually similar images but use them in a post-processing step instead of forming the initial queries. The goal is to have an additional constraint that the retrieved images should be close to each other as well as being close to the query image in the feature space.

Graph–theoretic approaches have been a popular tool in the computer vision literature, especially in object matching. Recently, graphs were used in image segmentation by treating pixels as nodes and some features as edge weights, and defining criteria like the normalized cut [7] and variations between intensity differences [4] to measure the disassociations between possible partitions of the graph. Graphs did not receive significant attention in image retrieval algorithms mainly due to the computational complexity of graph-related operations. Huet and Hancock [5] used attributed graphs to represent line patterns in images and used these graphs for image matching and retrieval.

The rest of the paper is organized as follows. The features used are discussed in Section 2. The new image retrieval algorithm is described in Section 3 and is followed by the summary of a graph–theoretic clustering algorithm in Section 4. Experiments and results are presented in Section 5. Finally, conclusions are given in Section 6.

2. Feature Extraction

In this work, we use the textural features that were described in [2, 1]. The feature vector consists of two sets of features which are intended to perform a multi-scale texture analysis which is crucial for a compact representation in large databases containing different types of complex images.

The first set of features are computed from the line-angle-ratio statistics which is a texture histogram method that uses the spatial relationships between lines as well as the properties of their surroundings. Spatial relationships are represented by the angles between intersecting line pairs and properties of the surroundings are represented by the ratio of the mean gray levels inside and outside the regions spanned by those angles. The second set of features are the variances of gray level spatial dependencies and are computed from the co-occurrence matrices for different spatial relationships. Each component f in the 28-dimensional feature vector is normalized as $f' = F_f(f)$, where $F_f(\cdot)$ is the cumulative distribution function of that component. This makes f' a random variable uniformly distributed in the $[0, 1]$ interval.

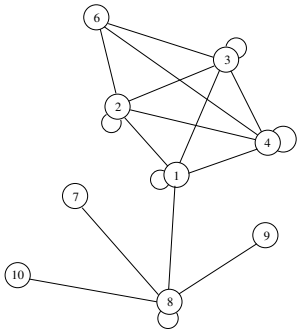
3. Image Retrieval

In most of the retrieval algorithms, a distance measure is used to rank the database images in ascending order of their distances to the query image, which is assumed to correspond to a descending order of similarity. In our previous work [2, 1] we defined a likelihood ratio to measure the relevancy of two images, one being the query image and one being a database image, so that image pairs which had a high likelihood value were classified as "relevant" and the ones which had a lower likelihood value were classified as "irrelevant". The distributions for the relevance and irrelevance classes were estimated from training sets and the likelihood values were used to rank the database images.

Unfortunately, none of the existing feature extraction algorithms can always map visually similar images to nearby locations in the feature space and it is not uncommon to retrieve images that are quite irrelevant to the query image simply because they are close to it. We believe that an efficient retrieval algorithm should be able to retrieve images that are not only close (similar) to the query image but also close (similar) to each other, and propose a new retrieval algorithm as follows. Assume we query the database and get back the best N matches. For each of these N matches we do a query and get back the best N matches again. Define S as the set containing the original query image and the images that are retrieved as the results of the above queries. S will contain $N^2 + 1$ images in the worst case. Then, we can construct a graph with the images in S as the nodes and can draw edges between each query image and each image in the retrieval set of that query image.

Query	Retrieved images				
1	1	2	4	3	8
2	2	1	3	4	6
3	3	2	1	4	6
4	4	1	3	2	6
5	5	7	2	6	10
6	6	2	3	4	5
7	7	5	8	10	9
8	8	7	9	10	1
9	9	8	7	4	10
10	10	7	8	9	5

(a) Results of all possible queries.



(b) Constructed graph when image 1 is the original query.

Figure 1: An example scenario for graph construction for a database of 10 images when $N = 5$.

We call these edges the set R where $R = \{(i, j) \in S \times S \mid \text{image } j \text{ is in the retrieval set when image } i \text{ is the query}\}$. An example graph is given in Figure 1. The feature vector distances between images, which correspond to two nodes that an edge connects, can also be assigned as a weight to that edge. We want to find the connected clusters of this graph (S, R) because they correspond to similar images. The clusters of interest are the ones that include the original query image. The problem now becomes finding P, where $P \subseteq S$ such that $P \times P \subseteq R$. This is called a clique of the graph. The clique with the largest number of nodes is called the major or maximal clique. The images that correspond to the nodes in P can then be retrieved as the final result of the query.

An additional thing to consider is that the graph (S, R) can have multiple clusters and some of these clusters can overlap. This is a desired property because image content is too complex to be grouped into distinct categories. Hence, an image can be consistent with multiple groups of images.

Additional measures are required to select the cluster that will be returned as the result of the query. In the next section we define the term "compactness" for a set of nodes. The cluster with the largest number of nodes can be retrieved as the final result. If more than one such cluster exists, we can select the one with the maximum compactness or we can compute the sum of the weights of the edges in each of those clusters and select the one with the minimum total weight.

This method increases the chance of retrieving similar images by not only ensuring that the retrieved images are close to the query image, but also adding another constraint that they should be close to each other in the feature space. In the next section we describe a graph–theoretic clustering algorithm which is used to find the clusters. Section 5 presents experimental results.

4. Graph–Theoretic Clustering

In the previous section, we proposed that cliques of the graph correspond to similar images. Instead of finding the cliques, to increase the speed, we use the algorithm described in Shapiro and Haralick [6] that finds "near-cliques" as dense regions instead of the maximally connected ones in the graph. To increase the speed further, the best N matches for the images in the database can be found offline so that graph clustering becomes the only overhead for a new query.

In the following sections, first we give some definitions, then we describe the algorithm for finding dense regions, and finally we present the algorithm for graph–theoretic clustering. The goal of this algorithm is to find regions in a graph, i.e. sets of nodes, which are not as dense as major cliques but are compact enough within user specified thresholds.

4.1. Definitions

- (S, R) represents a **graph** where S is the set of nodes and $R \subseteq S \times S$ is a symmetric binary relation on S.
- The **neighborhood** of X is defined as $\text{Neighborhood}(X) = \{Y \mid (X, Y) \in R\}$.
- **Conditional density** $D(Y|X)$ is the number of nodes in the neighborhood of X which have Y as a neighbor, $D(Y|X) = \#\{N \in S \mid (N, Y) \in R \text{ and } (X, N) \in R\}$.
- Given an integer K, a **dense region** Z around a node $X \in S$ is defined as $Z(X, K) = \{Y \in S \mid D(Y|X) \geq K\}$.
- $Z(X) = Z(X, J)$ is a **dense region candidate** around X where $J = \max\{K \mid \#Z(X, K) \geq K\}$ because if M is a major clique of size L, then $X, Y \in M$ implies that $D(Y|X) \geq L$. Thus $M \subseteq Z(X, L)$ and $K \leq L \leq \#Z(X, K)$.
- **Association** of a node X to a subset B of S is defined as

$$A(X|B) = \frac{\#\{\text{Neighborhood}(X) \cap B\}}{\#B} \quad (1)$$

  where $0 \leq A(X|B) \leq 1$.
- **Compactness** of a subset B of S is defined as

$$C(B) = \frac{1}{\#B} \sum_{X \in B} A(X|B) \quad (2)$$

  where $0 \leq C(B) \leq 1$.

4.2. Algorithm for Finding Dense Regions

A **dense region** B of the graph (S, R) should satisfy

1. $B = \{N \in Z(X) \mid A(N|Z(X)) \geq \text{MINASSOC}\}$ for some $X \in S$,
2. $C(B) \geq \text{MINCOMP}$,
3. $\#B \geq \text{MINSIZE}$

where MINASSOC, MINCOMP and MINSIZE are thresholds supplied by the user. To determine the dense region around a node X,

1. Compute $D(Y|X)$ for every other node Y in S.
2. Find a dense region candidate $Z(X, K')$ where $K' = \max\{K \mid \#\{Y|D(Y|X) \geq K\} \geq K\}$.
3. Remove the nodes with a low association from the candidate set. Iterate until all of the nodes have high enough association.
4. Check whether the remaining nodes have high enough average association (compactness) and whether the set is large enough.

When MINASSOC and MINCOMP are both 1, the resulting regions correspond to the cliques of the graph.

4.3. Algorithm for Graph Theoretic Clustering

Given dense regions, to find the clusters of the graph,

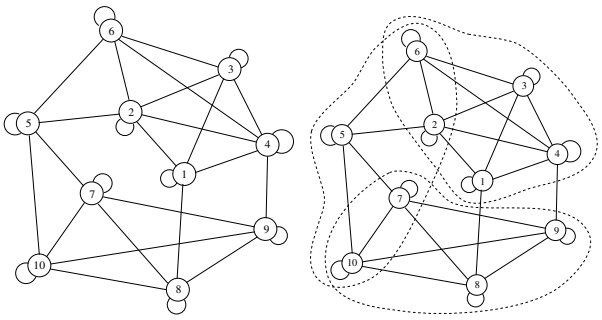
1. Define the **dense-region relation** F as

$$F = \{(B_1, B_2) \mid B_1, B_2 \text{ are dense regions of } R, \frac{\#B_1 \cap B_2}{\#B_1} \geq \text{MINOVERLAP or } \frac{\#B_1 \cap B_2}{\#B_2} \geq \text{MINOVERLAP}\} \quad (3)$$

   where MINOVERLAP is a threshold supplied by the user. Merge the regions that have enough overlap if all of the nodes in the set resulting after merging have high enough associations.
2. Iterate until no regions can be merged.

The result is a collection of clusters in the graph. Note that a node can be a member of multiple clusters because of the overlap allowed between them.

For the example graph in Figure 1, resulting cluster for image 1 is $\{1, 2, 4, 3, 6\}$. Image 6 is retrieved instead of image 8 because it is more consistent with the rest of the retrieved images.



(a) Graph for the whole database. (b) Example clusters are marked by dashed lines.

Figure 2: Example clusters of the graph when all images in Figure 1 are used as queries.

5. Experiments and Results

5.1. Database Population

The test set consists of 340 images which were randomly selected from a database of approximately 10,000 aerial and remote sensing images. To form the groundtruth for performance evaluation, these images were grouped into 7 categories; parking lots, roads, residential areas, landscapes, LANDSAT USA, DMSP North Pole and LANDSAT Chernobyl.

5.2. Clustering Experiments

The first step of testing the proposed retrieval algorithm is to check whether the clusters formed by the graph–theoretic clustering algorithm are visually consistent or not. First, each image is used as a query, and for each search, N top-ranked images are retrieved. Then, a graph for the whole database is constructed with all images as nodes and N edges corresponding to the N top-ranked images for each node. Some possible clusters for the example database of Figure 1 are given in Figure 2.

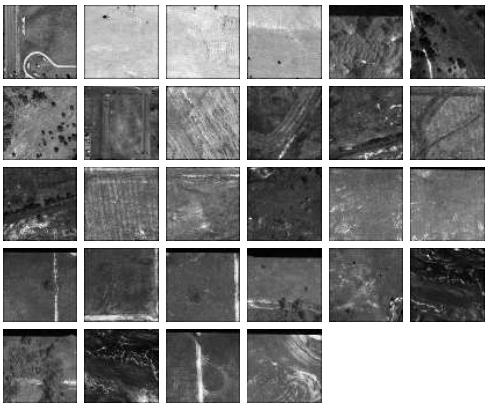
To evaluate the consistency of a cluster, we define

$$\textit{Consistency} = \frac{1}{K}\sum_{k=1}^{K}\frac{\#\{i \mid GT(i) = GT(k), i = 1, \ldots, K\}}{K} \tag{4}$$

where K is the number of images in the cluster and $GT(i)$ is the groundtruth group that image i belongs to. The term inside the summation indicates the percentage of the cluster that image k is correctly associated with.

In our experiments, MINSIZE and MINOVERLAP were fixed to be 12 and 0.80 respectively. 2220 clustering tests were performed for various values of $N \in [10, 100]$, MINCOMP $\in [0.3, 1.0]$ and MINASSOC $\in [0, \text{MINCOMP}]$. Maximum average *Consistency* of 0.6013 was obtained with the parameters $N = 24$, MINCOMP $= 0.5$ and MINASSOC $= 0.4$. Example clusters using these parameters are given in Figure 3. We obtained larger values for *Consistency* when we allowed some images to remain unclustered. For example, *Consistency* of 0.75 was obtained when only 6% of the images were unclustered. We observed that decreasing N or increasing MINCOMP or MINASSOC increases both *Consistency* and the number of unclustered images.

(a) *Consistency* $= 0.8347$.



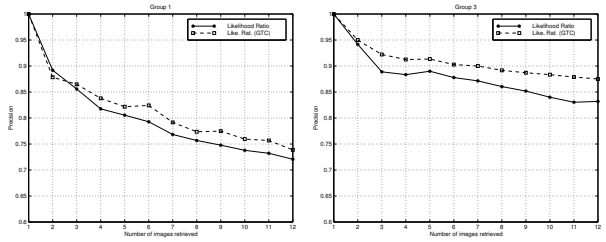
(b) *Consistency* $= 0.8674$.

Figure 3: Example clusters for N=24, MINCOMP=0.5, MINASSOC=0.4.

We also clustered the feature space using the k-means algorithm. The results are not presented here due to space limitations. For a given number of clusters, slightly lower consistencies were obtained, mainly because clusters were not allowed to overlap in this algorithm.

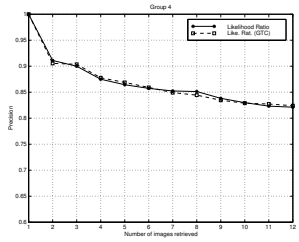
5.3. Retrieval Experiments

We performed experiments using all of the 340 groundtruth images as queries and retrieved images in the cluster with the largest number of nodes for each query. Results of the clustering experiments of the previous section were used to select the best parameter set to be used in retrieval. For comparison, we also retrieved only 12 top–

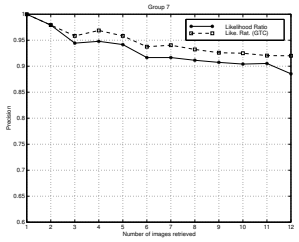


(a) Group 1

(b) Group 3



(c) Group 4



(d) Group 7

Figure 6: Precision as a function of the number of images retrieved. Solid lines correspond to retrievals using only the top–ranked images and dashed lines represent the graph–theoretic search.

ranked images (no clustering) for each query. When N is set to be equal to the number of images in the database, the graph reduces to a single clique and the graph–theoretic search becomes equivalent to retrieving N top–ranked images.

Example queries are given in Figures 4 and 5. We can observe that some images that are visually irrelevant to the query image can be eliminated after the graph–theoretic clustering. This is consistent with Figure 6 where average precision for some groundtruth groups are given. The average precision for the whole database was 0.76 (compared to 0.72 when only 12 top-ranked images are retrieved) which shows that approximately 9 of the 12 retrieved images and the query image belong to the same groundtruth group, i.e. the retrieved images are visally similar to the query image.

Since we use the top-ranked images to construct the graph, the initial precision before clustering should be large enough to prevent the graph being dominated by images visually irrelevant to the query image. In our experiments, a significant improvement to an average precision of 0.83 was observed when the initial precision was greater than 0.5. On the other hand, when the initial precision was less than 0.5, the average precision after clustering was 0.23.

The groundtruth used in the experiments and more retrieval examples can be found in our home page at http://isl.ee.washington.edu/∼aksoy/research/database.shtml.

6. Conclusions and Future Work

Image retrieval algorithms have a common assumption that images which are mapped to nearby locations in the feature space are always visually similar. However, it is not uncommon to retrieve images that are quite irrelevant to the query image simply because they are close to it in the feature space. In this paper we addressed this problem by introducing a graph–theoretic approach for image retrieval by formulating the database search as a problem of finding the cliques of a graph that is constructed from the top-ranked results of successive queries.

Experiments showed that some images that are visually irrelevant to the query image can be eliminated after the graph–theoretic clustering. This graph clustering approach does not directly depend on the feature extraction algorithm and is used as a post-processing step in retrieval. The improvements in features will also improve the results of clustering. Possible future work include feature selection to find the images that are inconsistent with other images in the same cluster and to check which features caused them to be close to the rest of the images and which features can distinguish them.

References

[1] S. Aksoy. Textural features for content-based image database retrieval. Master's thesis, University of Washington, Seattle, WA, June 1998. Online: http://isl.ee.washington.edu/∼aksoy/thesis.shtml.

[2] S. Aksoy and R. M. Haralick. Textural features for image database retrieval. In *Proceedings of IEEE Workshop on Content-Based Access of Image and Video Libraries, in conjunction with CVPR'98*, pages 45–49, Santa Barbara, CA, June 1998.

[3] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using EM and its application to image querying and classification. *submitted to PAMI*.

[4] P. Felzenszwalb and D. Huttenlocher. Image segmentation using local variation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 98–104, Santa Barbara, CA, June 1998.

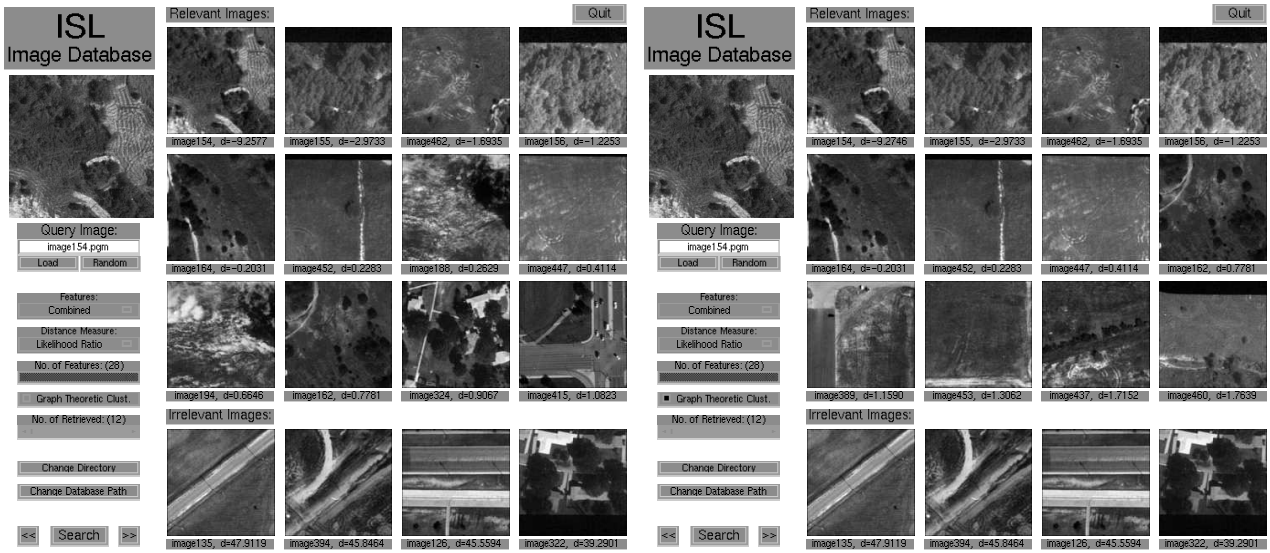
[5] B. Huet and E. Hancock. Fuzzy relational distance for large-scale object recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 138–143, Santa Barbara, CA, June 1998.

[6] L. G. Shapiro and R. M. Haralick. Decomposition of two-dimensional shapes by graph-theoretic clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(1):10–20, January 1979.

[7] J. Shi and J. Malik. Normalized cuts and image segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 731–737, Puerto Rico, June 1997.

(a) Using only 12 top–ranked images. (b) Using graph–theoretic clustering.

Figure 4: Example query 1. Upper left image is the query. Among the retrieved images, first three rows show the 12 most relevant ones in descending order of similarity and the last row shows the 4 most irrelevant ones in descending order of dissimilarity. When clustering is used, only 12 of the images that have the smallest distance to the original query image are displayed if the cluster size is greater than 12.



(a) Using only 12 top–ranked images. (b) Using graph–theoretic clustering.

Figure 5: Example query 2.