Bayesian Feedback in Data Clustering

A. K. Jain, Pavan K. Mallapragada and Martin Law
Department of Computer Science and Engineering
Michigan State University, East Lansing, MI-48824
{*jain,pavanm,lawhiu*}*@cse.msu.edu*

Abstract

In many clustering applications, the user has some vague notion of the number and membership of the desired clusters. However, it is difficult for the user to provide such knowledge explicitly in the clustering process. We propose a solution to circumvent this difficulty by introducing a feedback mechanism. The notion of Bayesian inference for relevance feedback in content-based image retrieval is modified for data clustering. Given the number of clusters, the proposed algorithm seeks information about the target partition by asking the user a sequence of queries about whether a pair of objects should be put in the same cluster or not. Information-theoretic criteria is adopted to select the queries to be presented to the user. The assumption made here is that cluster labels are "smooth", i.e., similar objects should share the same cluster labels. We show that it is possible to obtain reasonable partitions based on the user feedback alone, without the need of specifying a clustering objective function.

1 Introduction

The goal of clustering is to discover the "natural" grouping of a data set. Clustering is known to be ill-defined, and different clustering algorithms [3] impose different cluster structure on the data. Typically, the user needs to select an appropriate representation and a clustering algorithm that match the given data. This is very difficult because it is non-trivial to relate the "low-level" aspects of clustering such as the clustering objective function to the "high-level" semantics of the desired cluster structure. For example, suppose the user wants to cluster a set of face images into male and female clusters. This notion of male and female is based on the understanding of the image, and it is an example of high-level semantics. To conduct the clustering, the user chooses a representation (which can be eigenface coefficients in this example), and selects a clustering algorithm by making an educated guess on the properties of the desirable clusters. If the user suspects that the clusters are roughly hyperspherical, the k-means algorithm can be used. This approach, however, is far from ideal, because the fact that the user is seeking male/female clusters is used only implicitly to specify the clustering process.

A similar problem illustrating the difference between high-level semantics and low-level algorithm details appears in content-based image retrieval (CBIR) [11]. *Relevance feedback* has been suggested to circumvent this problem in CBIR by first identifying a small subset of candidate images and then asking the user to specify which of these images are relevant to the query. The answer provided by the user is used to refine the search, leading to a new set of retrieved images that are more likely to be relevant. There are three main approaches [8] for relevance feedback: (i) query vector modification modifies the low-level representation of the user's query, (ii) feature relevance estimation modifies the importance of different features in the low-level representation, and (iii) Bayesian inference estimates the posterior probability that an image is relevant conditioned on the feedbacks received from the user. These probabilities are used to locate the target image. The last approach, which we also call "Bayesian feedback", is particularly interesting, because it is principled and can be extended to other domains.

We propose to use the notion of relevance feedback to data clustering. Feedback is particularly appealing when clustering is used as an exploratory tool (as it should be), because the user is expected to be involved in understanding the structure of the data. We have modified the Bayesian inference for data clustering by replacing the target image by the target partition. Instead of asking the user to provide a clustering objective function, the system solicits feedback from the user by asking the user to state if two particular objects belong to the same cluster or not. Information-theoretic criteria are used to identify the successive pairs of objects to be presented to the user automatically. The user's feedback is then used to infer the most likely partition of the data. We assume that the number of clusters are known and, the cluster labels are "smooth". This smoothness require-

ment is specified by how likely the cluster label of an object is replaced by the cluster label of its neighbors, and it can be derived from a similarity measure between the objects. The smoothness information and the number of clusters are the only input required by the proposed algorithm. While relevance feedback in clustering was used in [1], the approach there is heuristic and modifies the features weights by some pre-specified amounts.

2 Bayesian Relevance Feedback

The goal of Bayesian relevance feedback is to identify a target state Z out of a set of possible states $\mathcal{Z}$ based on the user feedback. We shall model Z as a random variable, whose prior distribution $P(Z = z)$ corresponds to the a priori belief that the target state is z without any feedback. Often, $P(Z = z)$ is a constant, meaning that there is no preference towards any particular state. Let $\mathcal{Q}$ denote the set of queries[1] the system can present to the user. The t-th query presented by the system to the user is denoted by $Q^{(t)}_{\boldsymbol{\theta}}$, where $\boldsymbol{\theta}$ is an index into $\mathcal{Q}$. For brevity, sometimes we drop the subscript and write $Q^{(t)}$. The response of the user to the query is modeled stochastically by treating $Q^{(t)}$ as a random variable. We assume the following conditional independence:

$$P(Q^{(1)} = a_1, \ldots, Q^{(t)} = a_t|Z) = \prod_{l=1}^{t} P(Q^{(l)} = a_l|Z). \quad (1)$$

Here, $Q^{(t)} = a_t$ means that the feedback by the user on the query $Q^{(t)}$ is a_t. The distribution $P(Q^{(t)} = a_t|Z = z)$ corresponds to how the user replies to $Q^{(t)}$ if the target state is z, and this is known as the "user model". Let R_t denote the history of feedback $R_t = \{Q^{(1)} = a_1, \ldots, Q^{(t)} = a_t\}$. The posterior distribution $P(Z = z|R_t)$ tells us how likely a particular state z is the target state, given the set of t feedbacks.

The second component of Bayesian relevance feedback is the *display model*, which describes how the system selects the query. The greedy approach of constructing a query that maximizes $P(Z = z|R_{t-1})$, while having the highest chance of locating the target state, is often suboptimal because the answer of such a query provides little new information. Instead, the system should construct a query whose answer is most useful. There are two possibilities here. The system can pick θ such that $H(Z|Q^{(t)}_\theta, R_{t-1})$ is the smallest, where $H(.)$ denotes the Shannon entropy. Intuitively, this means that the query whose answer can minimize the uncertainty of the target state, on average, is selected. Because

$$H(Z|R_{t-1}, Q^{(t)}_\theta) = H(Z|R_{t-1}) - I(Z; Q^{(t)}_\theta|R_{t-1}),$$

this minimization is equivalent to finding $\boldsymbol{\theta}$ that maximizes $I(Z; Q^{(t)}_\theta|R_{t-1})$. The second possibility is to present a query where the system is most uncertain about its answer. This means that the system selects $\boldsymbol{\theta}$ that maximizes $H(Q^{(t)}_\theta|R_{t-1})$. If the knowledge $Z = z$ determines the answer to $Q^{(t)}_\theta$ for all θ, $H(Q^{(t)}_\theta|Z, R_{t-1})$ is zero, and these two approaches become equivalent.

[1] In some CBIR systems, the initial image provided by the user to begin the search is also known as the "query". In this paper, "query" refers to the question presented by the system to the user in order to solicit feedback.

3 Bayesian Feedback In Clustering

Suppose we want to cluster a set of n objects into K clusters. Each data partition can be represented by the vector $z = (z_1, \ldots, z_n)$, where z_i denotes the label of the i-th object, and $z_i \in \{1, \ldots, K\}$. The state space $\mathcal{Z}$ is thus $\{1, \ldots, K\}^n$. Note that this formulation ignores the label permutation problem in clustering: for any permutation σ of 1 to K, $\mathbf{z}$ and $\sigma(\mathbf{z})$ correspond to the same partition. This, however, does not create a difficulty for Bayesian relevance feedback as long as $P(Z = \mathbf{z}|Q^{(t)}) = P(Z = \sigma(\mathbf{z})|Q^{(t)})$. Let Z_i denote the i-th component of Z, i.e., it is the random variable for the cluster label of the i-th object.

The prior $P(Z)$ can be set to a constant as described in Section 2. Alternatively, the user can begin with a clustering objective function $f(\mathbf{z}; \mathcal{Y})$, where a small value of $f(\mathbf{z}; \mathcal{Y})$ denotes a good cluster structure. Note that $f(\mathbf{z}; \mathcal{Y}) = f(\sigma(\mathbf{z}); \mathcal{Y})$ for all σ. Examples of $f(.)$ that can be used include graph-cut [7], normalized cut [6], or the sum of square error used. The prior distribution in this case can be written as

$$p(Z = \mathbf{z}) \propto \exp\left(-\rho f(\mathbf{z}; \mathcal{Y})\right).$$

Here, ρ is a parameter that controls the strength of the prior. In our experiments, we have set ρ to 0, meaning that the clusters are only obtained based on the feedback.

Inspired by recent work on clustering with instance-level constraints (see [4, 10, 5], and the references therein), a natural query to be presented to the user is: "should the u-th and the v-th objects be put in the same cluster"? Because of this dependence on u and v, we shall write the query as $Q^{(t)}_{uv}$. The answer to this query can either be "yes" (represented by 1) or "no" (represented by 0), and this effectively imposes a constraint on the cluster labels of the u-th and the v-th objects.

The next step is to specify the user model $P(Q^{(t)}_{uv} = 1|Z = \mathbf{z})$. At first glance, one may want to define this distribution based on z_u and z_v, the cluster labels of the u-th and the v-th objects. The answer of the query leads to a constraint on z_u and z_v. However, this will lead to the "non-smooth" cluster label problem mentioned in [9]: objects that are similar can have different cluster labels. To overcome this, we adopt the solution in [9] and consider the "average" label X_u, which can be viewed as the smoothed version of the cluster labels of the objects that are similar

to the u-th object. X_u is a random variable that takes value from 1 to K, and

$$P(X_u = l|Z = \mathbf{z}) = \sum_j \frac{s_{uj}}{\sum_j s_{uj}} I(z_j = l) \quad (2)$$

Here, s_{uj} denotes a non-negative similarity between the u-th object and the v-th object. For convenience, the similarities can be normalized such that $\sum_j s_{uj} = 1$, modifying Eq (2) as $P(X_u = l|Z = \mathbf{z}) = \sum_j s_{uj} I(z_j = l)$. Defining the user model based on X_u and X_v has the effect of imposing a constraint on the average label of the u-th and the v-th objects. The use of average label automatically propagates the effect of this constraint to the neighbors of the u-th and the v-th objects. From a generative viewpoint, X_u obtains its value by first selecting the j-th object with the probability $s_{uj}/\sum_j s_{uj}$. After that, X_u assumes the cluster label of the selected object. We define a loss function in terms of the user feedback at iteration t, $a_t = \{+1, -1\}$ and the cluster labels Z as,

$$l(a_t, \mathbf{z}) = a_t E[I(X_u = X_v)|Z = \mathbf{z}], \quad (3)$$

where $I(.)$ denotes the indicator function taking the value one if the argument is true, and zero otherwise. The loss function for the answers is the sum of individual losses, i.e $l(\mathbf{a}, \mathbf{z}) = \sum_{t=1}^{T} a_t E[I(X_u = X_v)|Z]$, where, $\mathbf{a} = \{a_1, \cdots, a_T\}$. Using the loss function, we define the user model as $P(Q^{(t)} = a_t|Z = \mathbf{z}) \propto \exp\{\lambda l(\mathbf{a}, \mathbf{z})\}$ where λ is a parameter controlling the strength of the effect of the loss. We have the posterior $p(Z = \mathbf{z}|R_t)$

$$p(Z = \mathbf{z}|R_t) \propto \exp\big(-\rho f(\mathbf{z}; \mathcal{Y}) + \lambda l(\mathbf{a}, \mathbf{z})\big) \quad (4)$$

To determine the query to be presented, a pair of points indexed by (u, v), has to be selected from the dataset. We use an entropy based objective to select the pair (u, v). Intuitively, this corresponds to selecting the pair of points on which the system is most uncertain about their cluster labels being the same, i.e.,

$$(u, v)^* = \arg\max_{u,v} H(I(X_u = X_v)|Z). \quad (5)$$

Computing the conditional entropy involves a summation of n^K terms. We use an approximation for the entropy, which is presented in the next section.

4 Mean Field Approximation

The posterior distribution defined in Eq (4) is difficult to analyze, because computation of the normalization constant involves a summation of n^K terms, which is intractable for any reasonable value of n. This also creates difficulty in computing the expectation in Eq (5). Therefore, we resort to the mean field approximation.

We approximate the posterior $P(Z|R_t)$ using a distribution $\Phi(Z)$, where $\Phi(Z) = \prod_{i=1}^{n} \phi_i(z_i)$. The goal here is to minimize the KL-divergence between $\Phi(Z)$ and $P(Z|R_t)$ to obtain the $\phi_i(k)$, for $i = 1, \cdots, n$; $k = 1, \cdots, K$. Each $\phi_i(.)$ is essentially a distribution over the cluster labels of a sample, and $\sum_k \phi_i(k) = 1$. Formally, the objective can be stated as a minimization of $J(\Phi)$, where

$$J(\Phi) = \sum_{k=1}^{K} \sum_{i=1}^{n} \phi_i(k) \log \phi_i(k) - \lambda E_\Phi[l(\mathbf{a}, \mathbf{z})] \quad (6)$$

under the constraints $\sum_k \phi_i(k) = 1$. The formulation is permutation invariant, and we break this invariance by conditioning on $Z_1 = 1$. Using the Lagrange multiplier method, a solution for $\phi_i(k)$ is obtained as

$$\phi_i(k) = \frac{\exp\{\lambda g(\mathcal{S}, i, k, \Phi)\}}{\exp \sum_k \{\lambda g(\mathcal{S}, i, k, \Phi)\}}, \quad (7)$$

where $g(\mathcal{S}, i, k, \phi) = \sum_{\substack{j \neq 1 \\ j \neq i}} \gamma_{ij} \phi_j(k) + \gamma_{1j} \phi_j(k) I(k = 1)$, and $\gamma_{ij} = s_{ui} s_{vj} + s_{uj} s_{vi}$. Note that there is no closed form solution for each $\phi_i(.)$, and hence an iterative scheme is adopted. We start with a random initialization of $\phi(.)$ and iterate using Eq (7). The solution presented in Eq (7) is valid only in the case of two clusters, however the expression has been extended for multiple clusters. A maximum aposteriori inference on the distribution $\Phi(Z)$ is used to obtain the individual cluster labels of the points,

$$z_i = \arg\max_j \phi_i(j). \quad (8)$$

The distribution $\Phi(Z)$ is also useful in the computation of the entropy for the display model. The entropy computation in Eq (5) requires $P(X_u = X_v|Z)$, which is approximated using $\Phi(Z)$ as

$$P(X_u = X_v|Z) \approx \sum_{i \neq j, k} s_{ui} s_{vj} \phi_i(k) \phi_j(k) + \sum_i s_{ui} s_{vi}$$

The time complexity of the search for the query pair maximizing the entropy in Eq (5) is $O(n^2)$. For large datasets, this may be turn out to be expensive, and can be overcome by limiting the search to a random subset of pairs chosen uniformly. There are several pairs with high entropy at the beginning of the feedback iterations, and we choose a random pair from the set of feasible solutions.

The algorithm can now be summarized as follows: (a) Initialize $\phi_i^{(0)}(k)$ to random values, (b) pick a query pair $(u^{(t)}, v^{(t)})$ using Eq (5), (c) Using this query and iterating using Eq (7), obtain $\phi_i^{(t)}(.)$, for $i = 1, \cdots, n$. (d) Infer the cluster labels using Eq (8). (e) Using the current $\phi_i^{(t)}(.)$ and $\mathbf{z}$ as initial values, repeat steps (b) to (e), until the stopping criterion is met. Clustering is stopped

when both the following criteria are met: (1) The fraction of points whose cluster labels are of high confidence is above a certain threshold α, i.e., $\frac{1}{n}\sum_i[I(c_i > \theta)] > \alpha$, where $c_i = \min_{k \neq z_i}(\phi_i(z_i) - \phi_i(k))$, (2) There is no change in the clustering for η consecutive iterations. The parameters θ, α and η are chosen empirically.

5 Experiments and Results

The proposed approach was tested on a synthetic dataset and three UCI [2] datasets: Heart, Wdbc and Iris. The synthetic dataset has 2 classes, with a total of 500 points with 2 features. Heart dataset has 2 classes, 270 samples with 9 features. Wdbc is a two class dataset with 569 samples and 14 features. Iris is a 3 class dataset, with 150 samples with 4 features. We use a two dimensional PCA representation of Iris data for the experiments.

An illustration of the clustering process is shown on the synthetic two-moon data. In Figure 1 each row shows a pair of images: the current clustering (the left image) and the scatter plot of the class conditional distribution for class 1. At each point in the scatter plot, the color varying from blue(0) to red(1), corresponds to the probability that the point belongs to class 1. The solid lines show the queries where the user answered "Yes" and the dotted lines show the queries where the user answered "No". The figure summarizes a session with the user to obtain the clustering. The clustering is not meaningful until a sufficient number of queries is presented, and approaches the true partition of the data as iterations increase.

For the experiments, we used a similarity measure derived from the Euclidean distance: $s_{ij} = \exp\{-||y_i - y_j||^2/2\sigma^2\}$. The value of the parameter σ is determined by selecting the ν^{th} percentile of all the non-zero pairwise distances. The value of λ in Eq (7) is chosen to make the probabilities sufficiently spread between 0 and 1. A very small value of λ would result in all cluster probabilities closer to 0.5 and a very high value would result in the probabilities closer to either 0 or 1. We used a λ value of 100. We set the value of the prior effect ρ to zero, such that the clustering is based completely on the feedback of the user. The algorithm parameters (λ, σ), and the stopping criterion parameters (θ, α, η) are given the same value for all the datasets used here, and are presented in Table 1.

We used Normalized Mutual Information (NMI) as an evaluation measure for clustering. The value of NMI ranges from 0 to 1. A value of 0 means that the cluster labels are independent of the true labels, where as a value of 1 indicates a perfect match. We have compared the proposed semi-supervised algorithm with supervised and unsupervised techniques. We used K-means clustering, 1-Nearest Neighbor (1NN) classifier and Bayes Linear Classifier evaluated using a 10-fold cross validation for baseline compari-

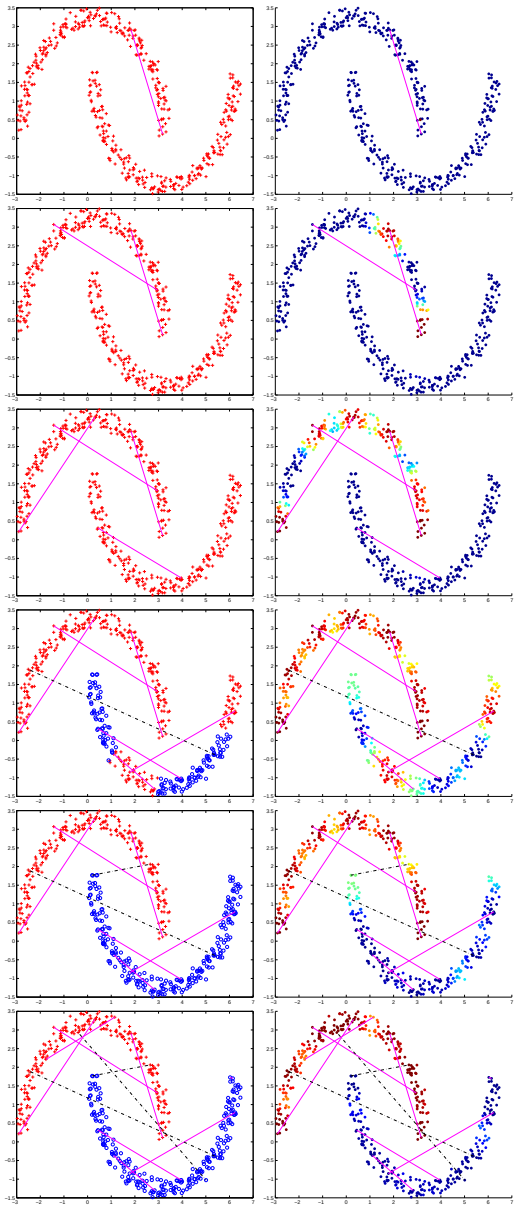

Figure 1. Clusterings and scatter plots over selected iterations (1,2,4,7,8,11) for the two-moon data. The algorithm converged after 11 iterations.

λ	ν	θ	α	η
100	20	0.1	0.85	3

Table 1. The parameter values used in the main algorithm (λ, ν) and the stopping criterion (θ,α,η).

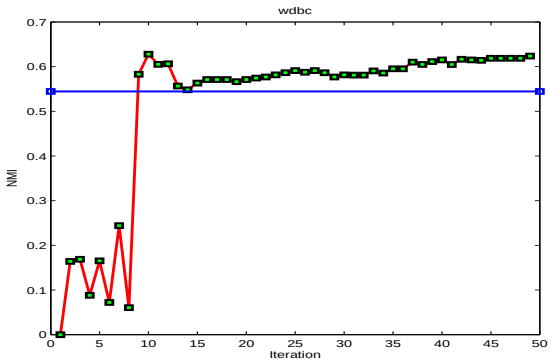

Figure 2. Improvement in NMI value on Wdbc dataset vs. iterations. The solid horizontal line shows the performance of the baseline unsupervised algorithm (K-means).

son. A summary of the NMI values for the K-means, LDA and the proposed algorithm is presented in Table 2. The last column in the Table 2 shows the number of queries presented by the system to the user. The % accuracies obtained corresponding to the NMI values in Table 2 are presented in Table 3. Figure 2 shows performance of the proposed algorithm over iterations, on the Wdbc dataset. For the initial iterations, the accuracy was low, and when sufficient number of queries are presented, the accuracy has shot up. The increase of performance over iterations is non-monotonic because of the presence of several local-minima in the solution space. We have attempted to overcome this by using multiple initializations for the mean-field approximation, which has alleviated the problem to certain extent. Results indicate that a significant improvement can be obtained from unsupervised learning using relevance feedback.

6 Conclusions and Future Work

We have proposed an approach to model the user's feedback in data clustering in the spirit of exploratory data analysis. We have presented a principled approach to obtain clustering based on user-feedback using Bayesian methods. No clustering objective function is specified in the proposed approach. Mean field approximation helps us to infer the intractable posterior distribution in an efficient manner. The query type and formulation can be further improved to include several data points instead of using only a pair of points. The query model and stopping criterion form the critical areas for further research. The algorithm also needs to be generalized and evaluated to handle more than two clusters ($K > 2$). Currently, the number of clusters is assumed to be known. It would be desirable to automatically learn the number of clusters.

Dataset	K-Means	1-NN	Linear	RF	#Queries
Synthetic	0.19	0.85	0.23	1.00	10
Heart	0.31	0.20	0.36	0.32	42
Wdbc	0.54	0.71	0.79	0.63	50
Iris	0.71	0.86	0.84	0.79	15

Table 2. Comparison of NMI values for K-means, 1-Nearest Neighbor, Bayes Linear Classifier, and the proposed Relevance Feedback (RF) algorithm. The last column shows the number of queries used by the proposed algorithm to achieve convergence.

Dataset	K-Means	1-NN	Linear	RF
Synthetic	75.00	97.80	77.40	100.00
Heart	80.60	75.92	84.07	81.48
Wdbc	90.78	95.25	96.66	91.03
Iris	78.33	96.00	95.33	88.67

Table 3. Comparison of accuracies (%) for K-means, 1-Nearest Neighbor, Bayes Linear Classifier and the RF algorithm.

References

[1] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. Technical Report TR2003-1892, Cornell University, 2003.

[2] C. B. D.J. Newman, S. Hettich and C. Merz. UCI repository of machine learning databases, 1998.

[3] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, Sept. 1999.

[4] B. Kulis, S. Basu, I. Dhillon, and R. Mooney. Semi-supervised graph clustering: A kernel approach. In *Proc. 22nd Intl. Conf. on Machine Learning*, pages 457–464, 2005.

[5] T. Lange, M. Law, A. Jain, and J. Buhmann. Learning with constrained and unlabelled data. In *Proc. Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 730–737, 2005.

[6] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on PAMI*, 22(8):888–905, 2000.

[7] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Trans. on PAMI*, 15(11):1101–1113, Nov. 1993.

[8] P.-Y. Yin, B. Bhanu, K.-C. Chang, and A. Dong. Integrating relevance feedback techniques for image retrieval using reinforcement learning. *IEEE Trans. on PAMI*, 27(10):1536–1551, Oct. 2005.

[9] S. Yu and J. Shi. Segmentation given partial grouping constraints. *IEEE Trans. on PAMI*, 26(2):173–183, 2004.

[10] Q. Zhao and D. Miller. Mixture modeling with pairwise, instance-level class constraints. *Neural Computation*, 17(11):2482–2507, Nov. 2005.

[11] X. Zhou and T. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia System*, 8(6):536–544, 2003.