IMPROVING IMAGE RETRIEVAL PERFORMANCE WITH NEGATIVE RELEVANCE FEEDBACK

Ashwin T. V.
IBM India Research Lab,
New Delhi - 110016, India.
vashwin@in.ibm.com.

Navendu Jain *
Computer Science and Engg. Dept.
Indian Institute of Technology,
New Delhi - 110016, India.

S. Ghosal
IBM India Research Lab,
New Delhi - 110016, India.
gsugata@in.ibm.com

ABSTRACT

Learning user perception of an image is a challenging issue in interactive content-based image retrieval (CBIR) systems. These systems employ relevance feedback mechanism to learn user perception in terms of a set of model-parameters and in turn iteratively improve the retrieval performance. Since the quantity of user feedback is expected to be small, learning the user's perception essentially involves parameter estimation with very few training points. We propose a novel, and more efficient method for relevance feedback in this paper. Contrary to existing geometric model-based relevance feedback methods, the proposed technique explicitly uses information about irrelevant data points to estimate the parameters of the model. This algorithm iteratively updates the parameters of the similarity metric so as to fit the relevant examples while excluding the irrelevant ones. This is achieved by modifying the weights associated with the relevant examples. Experiments on image and synthetic datasets demonstrate the retrieval effectiveness of the proposed approach.

1. INTRODUCTION

Popular content-based image retrieval (CBIR) systems employ relevance feedback techniques to capture the inherent subjectivity of user's perception. The prevalent idea is to assume a model describing the user's perception in a feature space, and iteratively refine the model-parameters based on user's preference on a set of currently retrieved images. The goodness of a relevance feedback algorithm may be measured in terms of how quickly (in less number of iterations) the relevant images can be retrieved from the database for a user.

Relevance feedback algorithms can broadly be classified as geometric, i.e., similarity metric-based (a distance metric is derived and used to retrieve database images closest to the query image) and probabilistic approaches [1](a probability distribution over the feature-space of images is derived so that the regions around relevant images have higher probability). Since the user cannot be expected to provide large quantity of feedback, the process of iteratively learning the user perception involves parameter estimation with a very few training samples. The focus of this paper is on a new similarity-based relevance feedback technique that explicitly uses information about irrelevant examples (according to a user).

Finding a similarity metric given a set of positive examples has been well studied in the past. The most commonly used similarity metric has been the quadratic distance or the *Mahalanobis Distance*, defined as,

$$d(x,\mu) = (x-\mu)^T Q(x-\mu) \text{ where } x,\mu \in R^d \ . \qquad (1)$$

In the image retrieval context, x and μ are feature vectors associated with two images. Given the user's preference on a set of images, the parameters μ and Q are estimated. μ can be considered as an estimate of the query the user has in mind and Q captures correlation between features. The approaches presented in the literature differ as to the type of the matrix Q that is assumed. MARS [2] assumed a diagonal Q and hence could not capture queries where two or more feature components are correlated as per user's perception. MindReader [3] using a full Q formulated the problem as that of estimating μ and Q to minimize the average distance of the relevant images from μ. In [4] Rui et al. propose a block diagonal Q to avoid the difficulties of estimating a full Q matrix from a small number of examples. In most of the above approaches only the relevant examples have been used to derive the similarity metric. MARS used negative examples to modify the learned query vector (μ).

The approaches discussed above require the user to specify weights for each relevant example to indicate its degree of relevance. This is not an easy task for the user, since assigning weights implies that the user be able to rank the examples. Recently automated techniques for obtaining these weights have been proposed These techniques require the user to specify that a particular example is either relevant or irrelevant, the system automatically chooses the weights based on this information. One such technique has been proposed in Hong [5]. The proposal is to train a Support Vector Machine (SVM) to classify the relevant feature vectors from the non-relevant ones. The output of the classifier for each relevant example is used as its weight. The output of the SVM classifier gives the distance of the input feature vector from the separating hyperplane in a transformed domain. Weighing the examples using this distance to estimate a quadratic metric may not be meaningful in the original space. Also, the authors do not address the issues associated with using a small training set obtained from the user's feedback in training a SVM classifier.

Our approach explicitly uses user-provided negative examples to weigh the relevant examples. The proposed algorithm begins with unit weights assigned to all the relevant examples. In each iteration, the parameters of a similarity metric are estimated using the current weights. The weights for the relevant examples are updated using their quadratic distances from the negative examples. The iteration is stopped when the estimated similarity metric

* Author performed the work as an intern at IBM India Research Lab.

encloses only relevant examples.

2. PROPOSED TECHNIQUE

2.1. Overview of the image retrieval system

The database consists of images and their corresponding feature vectors. The retrieval algortithm in our system uses a distance metric based similarity measure. In one iteration of relevance feedback, given an estimate of the target query and the distance metric, the distances of the feature vectors in the database from the estimated query are computed. The images in the database are ranked in increasing order of their distances. A fixed number of top ranked images are then shown to the user. The user then labels images which he considers relevant as *positive examples* and those segments which he feels unimportant or unacceptable as *negative examples*. The feature vectors corresponding to these images and their associated labels are used by the retrieval algorithm to obtain a new estimates of the target query and the distance metric. This completes the relevance feedback loop. The aim of the system is to maximize the number of relevant images retrieved using a small number of relevance feedback steps.

The retrieval algorithm uses the similarity metric proposed in Rui et al. [4]. This formulation is briefly described here. Let $x_i \in \mathcal{R}^p$ represent the feature vector for the i^{th} image. In many systems the features come from different classes (ex. shape, color, texture). Hence x_i can be expanded as $x_i = [x_{i1}, x_{i2}...x_{ic}]$. Here x_{ij} is a vector representing features from the j^{th} feature class, $|x_{ij}| = p_j$ the number of components in this class. Let X^r and X^n represent the set of relevant and non-relevant examples. $w^r \in \mathcal{R}^{|X^r|}$ represent the weights associated with the relevant examples. Let $\mathcal{D}$ represent the feature vectors in the database. The distance of an example $x \in \mathcal{D}$ given X^r, X^n and the weights w^r is computed as

$$d(x, X^r, w^r) = (x-\mu)^T Q (x-\mu) \tag{2}$$

where $\mu \in \mathcal{R}^p$ represents the target concept given by,

$$\mu(X^r, w^r) = \frac{\sum_{x_i \in X^r} w_i^r x_i}{\sum w_i^r}. \tag{3}$$

The $p \times p$ sized matrix Q is assumed to be block diagonal,

$$Q(X^r, w^r) = \begin{bmatrix} \pi_1 Q_1 & & & 0 \\ & \pi_2 Q_2 & & \\ & & \ddots & \\ 0 & & & \pi_c Q_c \end{bmatrix} \tag{4}$$

$$Q_i(X^r, w^r) = C_i^{-1}. \tag{5}$$

C_i is the covariance matrix for the i^{th} feature class, given by

$$C_i(X^r, w^r) = \frac{\sum_{x_j \in X^r} w_j (x_{ji} - \mu_i)(x_{ji} - \mu_i)^T}{\sum_{x_j \in X^r} w_j^r}. \tag{6}$$

Let $d_j(x_i) = (x_{ij} - \mu_j)^T Q_j (x_{ij} - \mu_j)$
The weights π_j for the feature classes are given by,

$$\pi_j = \frac{\sum_{i=1}^{|X^r|} w_i^r}{\sum_{i=1}^{|X^r|} w_i^r d_j(x_i)} \tag{7}$$

In the above formulation, the weights (w^r) are assumed to be provided by the user. As noted earlier this is not user friendly. We next describe our algorithm to determine these weights automatically.

2.2. Algorithm to assign weights to relevant examples

Our algorithm updates the weights (w^r) and the parameters of the similarity metric iteratively, so that the ellipsoids represented by the successive similarity metrics better capture the positive examples while excluding negative ones. The pseudo code is shown in Fig.(1). The algorithm begins by initializing all the weights w_i^r to one, i.e. initially all the relevant examples are considered to be equally important. In each iteration, the parameters of the similarity metric (μ and Q) using the current weight vector w^r are determined. The distances of the relevant and the non-relevant examples from the learned target concept (μ) are determined using (2). Let x_{max}^r denote the farthest positive example having a *non-zero weight* and d_{max}^r be its distance from μ. Let $\mathcal{E}$ be the ellipsoid defined by (μ, Q) and having a radius d_{max}^r. Let X_{new}^n represent the set of negative examples which fall inside the ellipsoid $\mathcal{E}$. The aim of the algorithm is to modify the parameters (μ, Q) in each iteration to reduce the number of such examples. This is achieved as follows. The weight of the farthest positive example x_{max}^r is set to 0. The weights of the other positive examples with non-zero weights are updated as the sum of their quadratic distances from the examples in X_{new}^n. The updated weights are then used to obtain a new estimate of the similarity metric parameters and the iteration proceeds. The iteration stops when the size of X_{new}^n becomes zero.

The algorithm proceeds by removing a positive example in each iteration. This positive example is considered an "outlier" since its inclusion in the estimation of the similarity metric results in negative examples (X_{new}^n) having smaller distances than positive examples. This would lead to the examples in X_{new}^n being retrieved again in the next iteration. To avoid this, the remaining relevant examples are weighted by their cumulative distances from examples in X_{new}^n. Hence, the metric estimated in the next iteration is forced away from X_{new}^n.

The algorithm stops when either of the following conditions are satisfied:

1. There exist no negative examples inside $\mathcal{E}$, i.e. X_{new}^n is empty. We have achieved our objective of determining the parameters (μ, Q) to best fit the set of relevant examples and excluding the irrelevant examples.
2. The number of *non-zero weighted* positive examples is so small that some of the matrices C_i in (4) become singular. This happens when the number of positive examples are too small or when they are distributed in the image space.

The algorithm is greedy in nature since the *farthest* positive example is removed in every iteration. Other methods to search for the best subset of relevant examples can also be employed. Jolion [6] describe a random sampling based approach.

3. EXPERIMENTS

The effectiveness of the proposed algorithm is experimentally demonstrated using artificially generated data and on images from Corel dataset. In the first experiment, we demonstrate how the algorithm learns the similarity metric on a synthetic dataset. The dataset

Input: X^r,X^n the relevant and non-relevant examples.
Output: Q and μ parameters of similarity metric.
Let min_d be the dimension of the feature
class with smallest number of components;
Let $w^r \in \mathcal{R}^{|X^r|}$; $w^r = 1$;
while (1) {
Calculate $Q(X^r, w^r)$ using (4);
Calculate $\mu(X^r, w^r)$ using (3);
$x^r_{max} = argmax_{\{x \in X^r\}}\{d(x, X^r, w^r)\}$
where d given by (2)
$d^r_{max} = d(x^r_{max}, X^r, w^r)$;
$X^n_{new} = \{x : x \in X^n, d(x, X^r, w^r) < d^r_{max}\}$;
if ($|X^n_{new}| = 0$) break;
$w^r[x^r_{max}] = 0$;
$w^r_i = \begin{cases} 0 & \text{if}(w^r_i = 0) \\ \sum_{y \in X^n_{new}} d(x^r_i, y) & \text{otherwise} \end{cases}$
$d(x, y) = (x - y)^T Q(x - y)$
if ($|\{w^r_i : w^r_i != 0\}| < min_d$)
break;
}
return Q and μ.

Fig. 1. Proposed algorithm for obtaining the similarity metric.

contains 24 relevant and 25 non-relevant examples. The ellipses representing the similarity metrics learned by the algorithm are as shown in Fig.(2). The ellipse learned in the first iteration encloses 11 non-relevant examples which reduces to zero after 13 iterations.

The aim of the second experiment is to demonstrate retrieval performance in a relevance feedback loop. We compare the performance of our algorithm with Rui's [4] method on an artificially generated dataset. The examples for the 2-d dataset (Fig.(3)) were generated uniformly inside {[0,1],[0,1]}. A randomly oriented ellipsoid centered at $(0.5, 0.5)$ was then generated. Examples inside the ellipsoid were then labelled as relevant. There are 189 relevant and 811 non-relevant examples. The experiment then simulates the relevance feedback loop. In the first iteration, the learning algorithm outputs a Euclidean metric centered at a randomly chosen point. The retrieved examples are labelled. The learning algorithm uses these labelled examples to output a similarity metric. This metric is then used to rank the examples in the dataset. A fixed number (100) of top ranked examples are chosen, their labels determined and input to the learning algorithm. Figs.(7) and (8) shows the ellipsoids representing similarity models learned by Rui's algorithm and our algorithm. It is clear that our algorithm is able to capture the relevant examples in the 6^{th} iteration, whereas even at the $9th$ iteration, Rui's algorithm captures only a part of the relevant examples. This can also be seen in the Fig.(4), which plots the number of relevant examples among the top 100 ranked examples by the two algorithms at successive iterations.

Our relevance feedback based retrieval algorithm was implemented in the iPURE framework [7],[8]. iPURE is a segmentation-based image retrieval system. We used 2000 images from Corel stock image dataset. The database is composed of features extracted from segments of these images. The feature vector representing a segment comprise of Color (LUV), position (centroid), size (number of pixels) and shape (shape moments). In each iteration of relevance feedback the user is shown images corresponding to 50 segments found to be most relevant by the retrieval algorithm. are shown in Figs.(5) and (6) compares the retrieval performance for category search of our algorithm with Rui's algorithm. Our algorithm shows rapid improvements over Rui's algorithm in the initial iterations, i.e. our algorithm captures the user's perception faster.

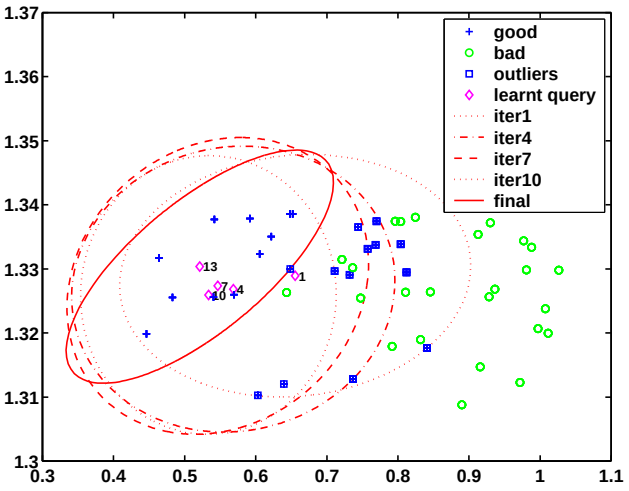

Fig. 2. Iterative learning of similarity metric on a synthetic 2-d dataset

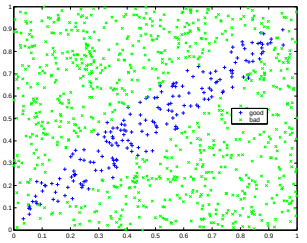

Fig. 3. Dataset to demonstrate improvement in relevance feedback loop.

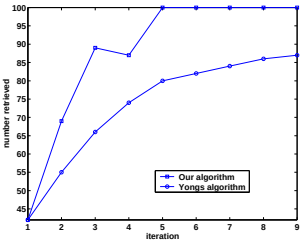

Fig. 4. Comparison of retrieval performance.

4. CONCLUSIONS

In this paper we propose a novel geometric similarity based relevance feedback technique to effectively incorporate irrelevant examples. Improvement in retrieval performance is demonstrated through experiments conducted on artificially generated and image datasets.

5. REFERENCES

[1] C. Meilhac and C. Nastar, "Relevance feedback and category search in image databases," in *IEEE Int. Conf. on Multimedia Computing and Systems*, 1999.

[2] Y. Rui, T.S. Huang, and S. Mehrotra, "Content-based image retrieval with relevance feedback in MARS," in *Proc. IEEE ICIP.*, 1997.

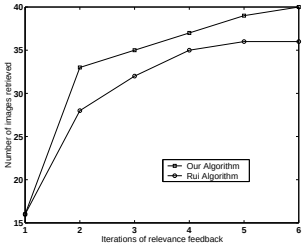

Fig. 5. Horse query

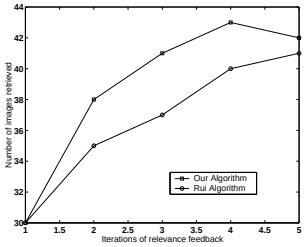

Fig. 6. Blue sky query

Comparison of retrieval performance for category search on Corel dataset.

[3] Y. Ishikawa, R. Subramanya, and C. Faloustos, "Mindreader: Query databases through multiple examples," in *Proc. of the 24th VLDB conference*, 1998.

[4] Y. Rui and T. Huang, "Optimizing learning in image retrieval," in *Proc. IEEE CVPR*, 2000.

[5] P. Hong, Q. Tian, and T. Huang, "Incorporate Support Vector Machines to content-based image retrieval with relevance feedback," in *Proc. IEEE ICIP*, 2000.

[6] J. M. Jolion, P. Meer, and S. Bataouche, "Robust clustering with applications in computer vision," *IEEE T PAMI*, vol. 13,NO. 8, 1991.

[7] G. Aggarwal, P. Dubey, S. Ghosal, A. Kulshreshta, and A. Sarkar, "iPURE: Perceptual and User-friendly REtrieval of images," in *Proc. IEEE Int. Conf. on Multimedia and Exposition*, 2000.

[8] G. Aggarwal, S. Ghosal, and P. Dubey, "Efficient query modification for image retrieval," in *Proc. IEEE CVPR*, 2000.

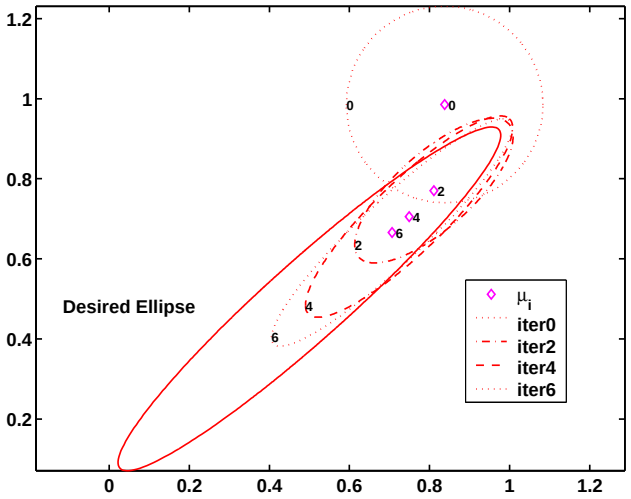

Fig. 7. Rui's algorithm.

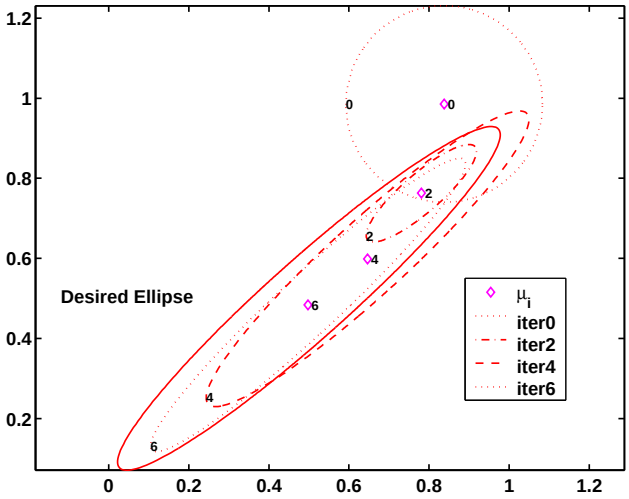

Fig. 8. Our algorithm.