Learning Feature Relevance and Similarity Metrics in Image Databases

Bir Bhanu, Jing Peng, and Shan Qing
Center for Research in Intelligent Systems
University of California, Riverside, CA 92521
{bhanu, jp, shan}@vislab.ucr.edu

Abstract

Most of the current image retrieval systems use "one-shot" queries to a database to retrieve similar images. Typically a K-NN (nearest neighbor) kind of algorithm is used where the weights of the features that are used to represent images remain fixed (or manually tweaked by the user) in the computation of a given similarity metric. However, neither all of the features are equally important for a given query nor a similarity metric is optimal for all kinds of images in a database. The manual adjustment of these weights and the selection of similarity metric are exhausting. Moreover, they require a very sophisticated user. In this paper we present a novel image retrieval system that continuously learns the weights of features and selects an appropriate similarity metric based on the user's feedback given as positive or negative image examples. Experimental results are presented that provide the objective evaluation of learning behavior of the system for image retrieval.

1. Introduction

The rapid advance in digital imaging technology makes possible the wide spread use of image libraries and databases. This in turn demands effective means for access to such databases. It is well known that simple textual annotations for images are often ambiguous and inadequate for image database search. Thus, retrieval based on image "content" becomes very attractive [1, 3]. Generally, s set of features (color, shape, texture, etc.) are extracted from an image to represent its content. Then the image database retrieval procedure becomes a K-NN search in the feature space under a given similarity metric.

There are three fundamental problems associated with this simple content-based image retrieval scheme. *First*, different similarity measures capture different aspects of perceptual similarity between images [3]. In general, what similarity metric to use is image dependent, and plays an important role in the outcome of image retrieval. *Second*, different features are unequal in their differential relevance for computing the similarity between images. When a user says that two images are similar, the user really means that the images are similar in an individual feature, some combination of the features, or some features still unknown to the user. This implies that the similarity does not vary with equal strength or in the same proportation in all directions in the feature space emanating from the query image. *Third*, the user understands more about the query, whereas the database system can only "guess" what the user is looking for during the retrieval process. The system must interact with the user to learn differential feature relevance and an appropriate similarity metric to guide its search and to iteratively refine its retrieval at run-time. In this paper, we present a novel method that enables image retrieval systems to learn differential relevance of features and optimal similarity metric in an efficient manner, and which is highly adaptive to query images.

1.1 Related Research

Ma and Manjunath [7] extract Gabor texture features from images and use a hybrid neural network to partition the feature space into clusters by training on a large number of labeled images. During testing a query image is assigned to a cluster label by the network and similar images from the same cluster are returned to the user. All features are treated equally important and Euclidean distance is used. Hu and Cercone [8] identify relevant features by using the rough set theory. Different weights are assigned to different features based on the significance value of features and the value-similarities. Both [7] and [8] do not involve user's feedback and can not refine their retrieval interactively. Minka and Picard [4] select features based

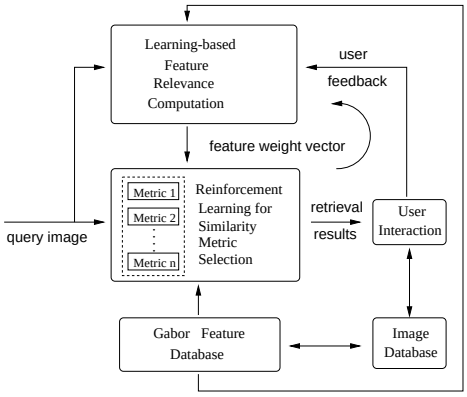


Figure 1. Learning features and similarity metrics.

on positive and negative examples to choose groupings for the query. This work is similar to [7] but feedback from the user is integrated for feature selection. However, all the selected features are treated with equal importance.

Unlike the above, in our research we select features based on user's feedback and where the query lies in the feature space. We also learn similarity metrics since different metrics are optimal for different classes of images [3]. Further, since feature relevance is strongly related to the similarity metric used, we develop an approach and show results where we learn both of them simultaneously.

2. System Overview

Figure 1 shows the overall architecture of our system. Images in the database are represented by the feature vectors whose components are normalized mean and standard deviations of responses from Gabor filters. The user presents a query image to the system. Since the system has no knowledge about the query at the beginning, it uses all the features (with equal relevance) to compute similarity measures and returns to the user highly similar images (based on the chosen metric for similarity) corresponding to the top K nearest neighbors in feature space of the image database. All the components of the feature relevance are initialized to the same value (1/dimension of the feature space) and are considered equally important for the first time retrieval. The user then marks the retrieved images as positive (e.g, click on an image by using the left mouse button) or negative (e.g, click on an image by using the right button). The user "thinks" that positive images look similar to the query image but the negative ones don't. These marked images constitute training data. From the query and training data, the system computes the relevance (weights) of different features. The components of the weight vector represent local relevance of each feature. They are adaptive to the location of the query image in the feature space. After the feature relevance has been determined, the system selects a similarity metric using reinforcement learning. Based on this new weight vector for feature relevance and the chosen similarity metric, the system restarts a new round of retrieval and repeat the above procedure until the user is satisfied with the results.

2.1 Local Feature Relevance

The performance of indexing for image databases can be characterized by two key factors. *First*, for a given query image, the relevance of all the features input to the indexing procedure may not be equal for retrieving similar images. Irrelevant features often hurt indexing performance. *Second*, feature relevance depends on where the query is made in the feature space. Capturing local relevance is essential for constructing successful indexing procedures in image databases.

In a two class (1/0) classification problem, the class label c at query $\mathbf{x}$ is treated as a random variable from the distribution $\{\Pr(1|\mathbf{x}), \Pr(0|\mathbf{x})\}$. c at $\mathbf{x}$ can be characterized by $y|_{\mathbf{x}} = \begin{cases} 1 & c|_{\mathbf{x}} = 1 \\ 0 & c|_{\mathbf{x}} \neq 1 \end{cases}$. This gives rise to

$$f(\mathbf{x}) \doteq \Pr(1|\mathbf{x}) = \Pr(y = 1|\mathbf{x}) = E(y|\mathbf{x}), \qquad (1)$$

To predict c at $\mathbf{x}$, $f(\mathbf{x})$ is estimated from a set of training data. In image retrieval, however, the "label" of $\mathbf{x}$ is known. All that is required is to exploit differential relevance of the input features to image retrieval. Consider the least-squares estimate for $f(\mathbf{x})$, given that $\mathbf{x}$ is known at dimension $x_i = z$, is

$$E[f|x_i = z] = \int f(\mathbf{x}) p(\mathbf{x}|x_i = z) d\mathbf{x}, \qquad (2)$$

where $p(\mathbf{x}|x_i = z)$ is the density of the other input features. Equation (2) shows the predictive strength (probability) once the value of just one (x_i) of the input features x_i is known. Then a feature relevance measure can be given by

$$r_i(\mathbf{z}) = (E[f|x_i = z_i])^2 / \sum_{l=1}^{q} (E[f|x_l = z_l])^2. \qquad (3)$$

One can see that $0 \leq r_i(\mathbf{z}) \leq 1$. $r_i(\mathbf{z}) = 0$ when $f(\mathbf{x})$ does not depend on x_i at query $\mathbf{z}$. And $r_i(\mathbf{z}) = 1$ when $f(\mathbf{x})$ depends entirely on x_i at $\mathbf{z}$. Values in between show the degrees of relevance that x_i exerts at $\mathbf{z}$. Thus, (3) can be used as a weight associated with each feature for weighted similarity metric computation.

In order to estimate (3), one must first compute (2). The retrieved images with feedback from the user can be used as training data to obtain estimates for (2) and (3). Let $\{\mathbf{x}_j, y_j\}_1^N$ be the training data, where $\mathbf{x}_j$ denotes the feature vector representing jth retrieved image, and y_j is either 1 or 0 marked by the user as a desired (positive or hit) or undesired (negative) image, respectively. Then, based on (1), one can estimate (2) according to

$$\hat{E}[y|x_i = z] = \sum_{j=1}^{N} y_j 1(|x_{ji} - z| \leq \delta) / \sum_{j=1}^{N} 1(|x_{ji} - z| \leq \delta)$$

where $1(\cdot)$ is an indicator function. δ can be chosen so that there are sufficient data for the estimation of (2). In addition, the above equation can be computed within a subregion, thus making the relevance measure more local. Note that this technique can be readily extended to multiple class situations where the user can grade retrieved images.

2.2 Learning Similarity Metrics

The particular framework adopted in this paper for selecting similarity metrics is connectionist reinforcement learning [5, 6]. Units in such a network are *Bernoulli quasilinear units* in that the output of such a unit is either 0 or 1, determined stochastically using the Bernoulli distribution with parameter $p_i = f(s_i)$, where f is the logistic function $f(s_i) = 1/(1 + \exp(-s_i))$, and $s_i = \sum_j w_{ij} x_j$ is the usual weighted summation of input values to that unit. For such a unit, p represents its probability of choosing 1 as its output value.

In the general reinforcement learning paradigm, the network generates an output pattern and the environment responds by providing the reinforcement r as its evaluation of that output pattern, which is then used to drive the weight changes according to the particular reinforcement learning algorithm being used by the network. For the Bernoulli quasilinear units used in this research, the REINFORCE algorithm prescribes weight increments equal to

$$\Delta w_{ij} = \alpha(r - b)(y_i - p_i)x_j \tag{4}$$

where α is a positive learning rate, b serves as a *reinforcement baseline*, x_j is the input to each Bernoulli unit, y_i is the output of the ith Bernoulli unit. In this paper, r is the ratio of positive hits against the number of retrieved images.

It can be shown [6] that, regardless of how b is computed, whenever it does not depend on the immediately received reinforcement value r, and when r is sent to all the units in the network, such an algorithm satisfies

$$E\{\Delta \mathbf{W}|\mathbf{W}\} = \alpha \nabla_{\mathbf{W}} E\{r|\mathbf{W}\} \tag{5}$$

where E denotes the expectation operator, $\mathbf{W}$ represents the weight matrix ($n \times (m+1)$, $m+1$ because of m inputs plus a bias, n is the number of units) of the network, and $\Delta \mathbf{W}$ is the change of the weight matrix. That is, the algorithm is guaranteed to converge to a local optimum. For adapting similarity metrics, it means that the process is moving towards selecting the similarity metric that increase the number of desired images retrieved.

We use a form of trial generating network in which all of the units are output units and there are no interconnections between them. This degenerate class of network corresponds to what is called a *team* of automata in the literature on stochastic learning automata. Each similarity metric is encoded by a set of Bernoulli quasilinear units and the output of each unit is binary as we have described earlier. Since we only consider three similarity metrics, the number of Bernoulli quasilinear units selected is two. For any Bernoulli quasilinear unit, the probability that it produces a 1 on any particular trial given the value of the weight matrix $\mathbf{W}$ is

$$Pr\{y_i = 1|\mathbf{W}\} = p_i = f(s_i) = \frac{1}{1 + e^{-s_i}}$$

where $s_i = \sum_j w_{ij} x_j$. Because all units pick their outputs independently, it follows that for such a team of Bernoulli quasilinear units the probability of any particular output vector $\mathbf{y}(t)$, corresponding to an instance of similarity metrics, conditioned on the current value of the weight matrix $\mathbf{W}$ is given by

$$Pr\{\mathbf{y}|\mathbf{W}\} = \prod_{i \in \{1, \cdots, n\}} p_i^{y_i} (1 - p_i)^{1 - y_i}. \tag{6}$$

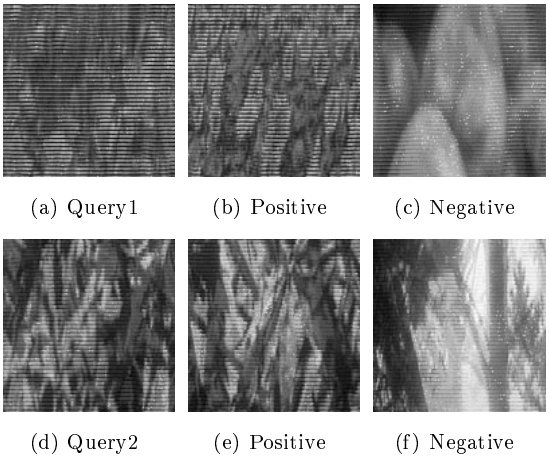
2.2.1 Similarity Metrics: The similarity metrics used in our system are defined as follows. Let $\mathbf{x}$ and $\mathbf{y}$ represent two feature vectors, then the similarity between $\mathbf{x}$ and $\mathbf{y}$ is defined as follows.

$$d_t(\mathbf{x}, \mathbf{y}) = \left[\sum_{i=1}^{q} r_i |x_i - y_i|^t \right]^{1/t} \tag{7}$$

where $\sum_{i=1}^{q} r_i = 1$, and they are computed using Eq. (3). Equation (7) yields weighted Euclidean metric ($t = 2$), city-block metric ($t = 1$), or dominance metric ($t = \infty$) [3]. We call these three metrics as metric 0, metric 1 and metric 2, respectively.

Table 1. Learning feature relevance for various similarity metrics.

	Euclidean Similarity Metric																									
Iters\Class	106	169	190	4	156	53	127	207	38	146	211	132	111	213	128	103	72	55	151	175	134	206	83	46	62	47	192	216
0	21	14	22	22	18	19	16	15	21	16	20	19	24	23	23	22	22	19	21	22	13	19	17	16	20	21	23	19
1	31	24	24	26	27	26	26	24	26	25	27	24	27	29	26	30	32	25	30	33	27	30	28	24	27	26	29	25
2		26	25					25				25												26				
3																												
	City-block Similarity Metric																									
Iters\Class	37	6	60	211	156	17	46	200	105	143	190	134	83	170	179	216	62	96	98	0	12	33	102	26	23	111	88	139
0	24	23	23	23	23	23	15	24	17	22	19	23	24	18	22	21	24	24	23	23	22	22	18	22	24	21	24	24
1	29	25	24	26	28	27	25	28	25	25	24	31	28	25	26	24	27	27	25	25	25	24	25	27	23	27	27	27
2			25								26					25						25			25			
3																												
	Dominance Similarity Metric																									
Iters\Class	136	169	159	157	37	114	52	55	211	164	156	65	33	165	49	38	188	93	211	26	30	111	188	211	105	5	175	26
0	19	14	20	15	23	18	23	20	20	15	19	17	20	24	21	15	21	7	17	19	9	14	15	9	17	19	23	16
1	32	29	26	24	25	32	33	28	26	24	30	25	25	26	24	31	27	25	23	27	25	23	28	18	27	25	30	22
2				25						29					25				26			27		27			26
3																												



(a) Query1 (b) Positive (c) Negative

(d) Query2 (e) Positive (f) Negative

Figure 2. Sample sets of database images.

2.3 Gabor Wavelet Based Features

All the images in our database are texture images. Gabor wavelets [9] are used to extract texture features. The feature database consists of feature vectors for all the images in the database.

A two dimensional Gabor function g(x,y) can be written as :

$$g(x,y) = \left(\frac{1}{2\pi\sigma_x\sigma_y}\right) exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)+2\pi j\omega_h x\right]$$

Using the above formula as the mother function, a set of self-similar filters are derived through the following generating function: $g_{mn}(x,y) = a^{-m}g(x',y')$, $x' = a^{-m}(xcos\theta+ysin\theta)$, $y' = a^{-m}(-xsin\theta+ycos\theta)$, where $\theta = n\pi/K$, n=0, 1, $\cdots$, K-1 and m=0, 1, $\cdots$, S-1. K is the total number of orientations and S is the total number of scales. $a = (\omega_h/\omega_l)^{1/(S-1)}$, where ω_h and ω_l are the highest and lowest center frequencies of the filter, respectively. σ_x and σ_y are the standard deviations along x and y dimensions.

Given an image I(x,y), its Gabor wavelet transform is then defined to be :

$$W_{mn}(x,y) = \int\int I(\alpha,\beta)g^*_{mn}(x-\alpha,y-\beta)d\alpha d\beta$$

where * indicates the complex conjugate. The mean μ_{mn} and the standard deviation σ_{mn} of the magnitude of the transform coefficients are used as feature components of a feature vector after being normalized by the standard deviations of the respective features over the entire feature database.

3. Experimental Results

• **Data and experiment:** There are 10,314 texture images of size 64 × 64 in our database that are represented by 48 dimensional feature vector. We use 24 Gabor filters (4 scales and 6 orientations) for feature extraction. For each filtered image, its normalized mean and standard deviation are used as features.

In order to perform the objective evaluation of our system, the images are partitioned into 220 classes. There are about 45 images in each class. These 220 classes form the *ground truth* for our experiments. For each query image, the system retrieves 40 images that are most similar to the query image based on its current similarity metric at each iteration. From the retrieved images, the system chooses those as positive examples that come from the same class as the query and uses rest of them as negative examples. In practice, these positive examples will be provided by the user. The feature relevance algorithm uses these positive and

Table 2. Typical results demonstrating simultaneous learning of features and similarity metrics. Table entries show (metric number, # of positive hits).

Iteration\ QueryClass	0	1	2	3	4	5	6	7
192	(2,24)	(1,21)	(2,23)	(2,24)	(2,24)	(0,25)		
98	(2,19)	(0,21)	(1,15)	(0,26)				
37	(0,24)	(2,24)	(0,25)					
217	(2,18)	(0,14)	(0,25)					
190	(0,23)	(0,23)	(0,24)	(2,13)	(2,19)	(2,23)	(2,21)	(0,25)
196	(1,20)	(2,35)						
52	(0,15)	(1,6)	(1,16)	(1,18)	(2,23)	(1,17)	(2,33)	
136	(1,18)	(0,26)						
164	(2,15)	(0,14)	(1,6)	(2,24)	(2,29)			
156	(2,19)	(2,30)						
49	(0,18)	(0,21)	(2,21)	(0,22)	(0,22)	(2,24)	(0,22)	(2,25)
207	(2,16)	(0,21)	(2,23)	(1,15)	(2,24)	(0,26)		
216	(0,24)	(2,19)	(0,24)	(1,24)	(1,28)			
145	(1,1)	(0,10)	(0,28)					
216	(0,23)	(1,26)						
85	(2,24)	(1,12)	(0,21)	(1,19)	(2,29)			
96	(2,21)	(2,23)	(2,25)					
131	(0,22)	(1,19)	(1,24)	(0,25)				
46	(1,14)	(2,15)	(1,21)	(0,15)	(0,25)			
177	(0,24)	(1,22)	(0,25)					

negative examples and the query image as input and generates an updated feature relevance to be used during the next iteration. Note that initially all features are equally relevant and the sum of relevance (weights) of all the features is unity. The termination conditions for the retrieval procedure are (i) the number of positive examples returned becomes larger than a certain threshold (25 in experiments), (ii) the number of iterations exceeds another threshold (20 in experiments). Two sample query images and corresponding positive and negative example images are shown in Fig. 2.

- **Optimality of similarity metrics:** We performed experiments to determine optimal similarity metric for each of the 220 classes. Similar to the results shown in [3], we find that different similarity metrics are optimal for different classes.
- **Learning feature relevance:** Table 1 shows sample retrieval results by using different similarity metrics where learning feature relevance improves performance over time. Three different similarity metrics are used. In each case, the first row indicates the classes from which queries come from. The second row shows the number of positive hits received after the first iteration. The third row shows the number of positive hits after the second iteration. Similarly, the results for other iterations are shown.
- **Learning feature relevance and similarity metrics:** Table 2 shows that our system can simultaneously learn both differential relevance of features and the selection of similarity metric. For example, for a query from class 217 the similarity metric selection changes from similarity 2 to similarity 0 to obtain 25 positive hits. Likewise, for a query from class 164 the number of positive hits increases from 15 to 29 using the same similarity metric (metric 2). Note that for two queries from class 216, initially metric 0 is selected and finally metric 1 provides the desired number of hits.

4. Conclusions

The paper shows that learning feature relevance and similarity metrics based on user's feedback can improve retrieval. The knowledge acquired during one retrieval can be gradually collected and it can become part of the database itself through continuous learning. Future work includes incorporating additional similarity measures and evaluation of the learning performance by designing human psychological experiments.

Acknowledgements: This work was supported by DARPA/AFOSR grant F49620-97-0184 at the University of California, Riverside. The contents of the information do not necessarily reflect the position or the policy of the Government.

References

[1] M. Flickner et al., "Query by image and video content: The QBIC system," *IEEE Computer,* Sept.1995.

[2] J.H. Friedman "Flexible metric nearest neighbor classification," *Tech. Report, Dept. of Statistics, Stanford University,* 1994.

[3] R. Jain, S.N.J. Murthy and L. Tran, "Similarity measures for image databases," *Proc. IEEE Conf. on Fuzzy Logic,* 1995.

[4] T.P. Minka and R.W. Picard, "Interactive learning with a "society of models"," *Pattern Recognition,* 30(4), pp. 565-81, Apr.1997,

[5] J. Peng and B. Bhanu, "Closed-Loop object recognition using reinforcement learning," *IEEE Trans. PAMI 20(2),* pp. 139-154, Feb. 1998.

[6] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning 8*, pp. 229-256, 1992.

[7] W.Y. Ma and B.S. Manjunath, "Texture features and learning similarity," *Proc. IEEE Conf. CVPR,* 1996.

[8] X. Hu and N. Cercone, "Rough sets similarity-based learning from databases," *Proc. Conf. on KDD,* pp. 162-167, 1995.

[9] X. Wu and B. Bhanu, "Gabor wavelet representation for 3-D object recognition," *IEEE Trans. on Image Processing*, 6(1), pp. 47-64, Jan. 1997.