Fast Active Tabu Search and its Application to Image Retrieval*

Chao Zhang[1], **Hongyu Li**[1], **Qiyong Guo**[2], **Jinyuan Jia**[1], **I-Fan Shen**[2]
[1]School of Software Engineering, Tongji University, Shanghai, China
[2]Department of Computer Science and Engineering, Fudan University, Shanghai, China
{325zhch,hyli,jyjia}@tongji.edu.cn, {qyguo,yfshen}@fudan.edu.cn

Abstract

This paper proposes a novel framework for image retrieval. The retrieval is treated as searching for an ordered cycle in an image database. The optimal cycle can be found by minimizing the geometric manifold entropy of images. The minimization is solved by the proposed method, fast active tabu search. Experimental results demonstrate the framework for image retrieval is feasible and quite promising.

1 Introduction

Content-based image retrieval (CBIR) has been an active research topic in multimedia for years. Compared to the text-based retrieval engines, CBIR performs retrieval in a geometric space rather than an image space. Although CBIR has achieved some success [Jing and Baluja, 2008], it is still far from satisfactory. The major difficulty of CBIR is the insufficiency in the description of images and semantic space. Recently, manifold based image representation approaches for image retrieval have drawn lots of attentions [Yu and Tian, 2006]. Most of these methods consider the image feature space as an embedded manifold and try to find the mapping between the feature space and the manifold. For instance, in [He *et al.*, 2004] a method was proposed to find an embedding of the image manifold where image retrieval was performed. In [Cai *et al.*, 2007], considered the retrieval problem as a classification problem on manifold and managed to learn a classification function on the image manifold. However, the properties of the mapping from low-level feature space to high-level semantic manifolds still remains unclear. In addition, the dimensionality of the semantic space is also unknown and hard to determine in advance. To avoid these problems, we propose a novel framework for image retrieval in this paper.

Geometric manifold entropy (GEOMEN) is the basis of the proposed framework. GEOMEN describes the connection and relevance between data. Our aim is to find an ordered cycle by minimizing GEOMEN. The minimization can be solved by fast active tabu search (FATS). FATS actually is the improved version of tabu search which is running on the GPU-based platform. Here the word 'active' means the search is intelligent and works along a meaningful direction. The proposed FATS method is quite efficient in solving the large-scale combinatorial optimization problems.

The following highlights the major contributions of the paper:

1. A new entropy function (GEOMEN) is defined to describe the connection and relevance between images.
2. The retrieval is treated as searching for an ordered cycle in an image database. The proposed framework not only works for image retrieval but also for other information retrieval problems, as long as significant features are extracted and employed.
3. Tabu search is a common solution to the optimization problems. However, picking up the best candidate in this method is very time consuming, especially for large-scale data sets. In this study, we improve it and propose a fast and intelligent method, named fast active tabu search. The main advantage of FATS is that it is very efficient for the large-scale optimization problems. Besides, FATS also can be applied to other related combinatorial optimization problems.

The rest of this paper is organized as follows: Section 2 discusses the definition of GEOMEN. We introduce the entropy minimization through active tabu search in Section 3. Section 4 presents the design of FATS on GPU. The framework for image retrieval is described in Section 5. Experimental results are presented in Section 6. Finally, we conclude the paper in Section 7.

2 Geometric Manifold Entropy

The representation of geometric manifolds is the core of the proposed method as well as the key to the success of retrieval. In this study, we make use of the spatial position of data points and local discrete curvature of manifolds as geometric representation of manifolds. This kind of representation is called geometric manifold entropy, for short, GEOMEN.

In particular, given a set of unorganized data in an m-dimensional space $X = \{x_i | x_i \in R^m, i = 1, 2, \ldots, n\}$, we first define a *cycle* of length n as a closed path without self-intersections. Each datum in this cycle is connected with

*This work was partially supported by STCSM (08511501001) and by 863 project (2007AA01Z176).

two neighbors and the corresponding connection order O is symbolized as

$$O = (o_1, o_2, \dots, o_n, o_1),$$

where the entry corresponds to the index of data. Then, the GEOMEN of the set X with the order O is represented as the sum of two components: spatial position $P(X, O)$ and geometric $G(X, O)$ as,

$$\mathcal{S}(X, O) = P(X, O) + G(X, O). \tag{1}$$

GEOMEN represents the smoothness and sharpness of the cycle with the connection order O. In addition, it is also a metric of disorder and similarity of the data in the embedded manifolds. Since manifold ranking can be thought as the problem of extracting a 1-dimensional manifold, actually a curve, we only consider the representation of GEOMEN on 1-dimensional curves.

If the embedded manifold is a 1-dimensional curve, the spatial component of GEOMEN is measured by the Euclidean distance,

$$P(X, O) = \frac{1}{n} \sum_{(i,j) \in O} d^2(x_i, x_j). \tag{2}$$

Here $d(x_i, x_j) = \|x_i - x_j\|$ represents the Euclidean distance between x_i and x_j and symbol (i, j) means that points x_i and x_j are connected in the cycle with the order O. At this time, the geometric component of GEOMEN is composed of two terms: the curvature κ of curves and a regularization term ρ. This can be formularized as:

$$G(X, O) = \frac{1}{n} \sum_{(i,j,k) \in O} \kappa^2(x_i, x_j, x_k) + \frac{1}{n} \sum_{(i,j,k,l) \in O} \rho^2(x_i, x_j, x_k, x_l). \tag{3}$$

where (i, j, k) and (i, j, k, l) has the same meaning to (i, j).

In the continuous domain, the curvature of a smooth curve is defined as the curvature of its osculation circle at each point. In the discrete domain, we measure the curvature κ of data at a particular point x_j in the following manner:

$$\kappa(x_i, x_j, x_k) = \|\lambda(x_i, x_j) - \lambda(x_j, x_k)\|. \tag{4}$$

In addition, since the discrete curvature is sensitive to noise, to improve the robustness of our algorithm, we introduce in the geometric component the regularization term:

$$\rho(x_i, x_j, x_k, x_l) = \|(\lambda(x_i, x_j) - \lambda(x_j, x_k)) - (\lambda(x_j, x_k) - \lambda(x_k, x_l))\|. \tag{5}$$

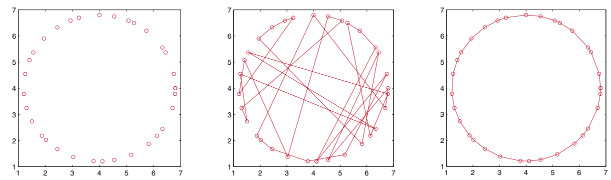
In Equations (4) and (5) the symbol $\lambda(x_i, x_j)$ means

$$\lambda(x_i, x_j) = \frac{x_i - x_j}{d(x_i, x_j)}.$$

3 Entropy Minimization via Active Tabu Search

From the definition of GEOMEN, if the data points are ordered enough, the entropy is supposed to be quite small. For instance, if a set of data distribute along a circle, as shown in Figure 1(a), it is clear that the entropy is the smallest if and only if the order is along the circle, as shown in Figure 1(c). Other possible combinations, e.g. the cycle shown in Figure 1(b), incline towards the more disorder and the higher entropy values. Therefore, with the purpose of finding the most ordered cycle with the minimal entropy, we need to minimize the GEOMEN,

$$O^* = \arg\min_{o} \mathcal{S}(X, O). \tag{6}$$



(a) Input data (b) Random cycle (c) Optimal cycle

Figure 1: Finding the most ordered cycle through entropy minimization. (a): the input data sampled from a circle; (b): a random order with a higher entropy; (c): the optimal cycle with the minimal entropy 0.384314.

3.1 Active Tabu Search

Finding a globally optimal solution to Equation (6) is a NP problem and completely impossible in practice as there exists $\frac{(n-1)!}{2}$ possible combinations in a cycle with n(n≥3) points. In this study, we approximate the global minimum of the entropy through the proposed active tabu search, which is an extension of the original tabu search [Glover, 1986] with the active learning technology [Ertekin *et al.*, 2007]. This method makes the search more intelligent and efficient by a narrowing search space. In our method, the neighborhood map and the tabu list H need to be designed.

The neighborhood map means the transformation from a cycle to another. It involves three aspects: 1) swapping a pair of points in the cycle; 2) shifting a point to other positions in the cycle; 3) inversing a specific fragment in the cycle. For instance, given a cycle containing n points with the order

$$O = \{o_1, o_2, o_3, \dots, o_n, o_1\},$$

we can obtain a new cycle with the order

$$O = \{o_4, o_2, o_3, o_1, o_5, \dots, o_n, o_4\},$$

by swapping o_1 and o_4, the order

$$O = \{o_2, o_3, o_4, o_1, o_5, \dots, o_n, o_2\},$$

by shifting o_1 to the position between o_4 and o_5, or the order

$$O = \{o_4, o_3, o_2, o_1, o_5, \dots, o_n, o_4\},$$

by inversing the fragment from o_1 to o_4.

The tabu list H is a short-term memory containing the representation of transformations. It is used to prevent previous transformations from being repeated. That is, the transformation in H is not permitted for a while in succeeding iterations. The short time is described by the variable *tabu-time*. When a new transformation is added to H, its tabu-time is initialized to be zero.

All elements, from the neighborhood set but not in the tabu list H, constitute a candidate subset CSS. By searching in the CSS, we can get the best candidate O_{best} with the minimal entropy as the optimal transformation in this iteration. Before complete one-time iteration, we need to increase the tabu-time of each transformation and remove those transformations whose tabu-time has reached the limitation from H. After that, the corresponding transformation of O_{best} needs to be added to H. For example, if O_{best} is obtained by swapping o_i and o_j, we add swapping (o_i, o_j) to H.

However, picking up O_{best} is very time-consuming, especially for large-scale data sets, since it could involve a full search in CSS. We employ the active learning technology [Ertekin *et al.*, 2007], which aims to reduce the labor cost of learning and select the most informative sample. As such, we do not need to search in the entire set CSS, but a randomly chosen subset L with constant size. In this case, $\sharp L \ll \sharp CSS$, where notion $'\sharp'$ stands for the size. We assume that two conditions must be held: 1) O_{best} chosen from L is among the top $p\%$ best in CSS with the probability $\eta\%$; 2) the probability that at least one of the candidates in L is among the top $p\%$ best is $1-(1-p\%)^{\sharp L}$. Therefore $\sharp L$ can be computed in terms of $p\%$ and $\eta\%$:

$$\sharp L = \left\lceil \frac{\log(1-\eta\%)}{\log(1-p\%)} \right\rceil, \quad (7)$$

where $\sharp L$ is obviously independent of $\sharp CSS$. In addition, an operation called *aberrance* is employed in order to make our method avoid falling into local optimum. That is, when the entropy is locally convergent, we randomly pick up a subset of CSS to disorder the current order O_{cur} directly. The time of this operation is restricted by the variable *aberrance-time*.

Given a set of data X, the active tabu search can be briefly described as follows:

Step 1 Initialize the tabu-time and the aberrance-time with constant numbers as their limitations, and the cycle of input data with a random order O, and the tabu list H with an empty set. Set $O_{cur} = O$, the current entropy $\mathcal{S}_{cur} = \mathcal{S}(X, O)$, the optimal order $O_{opt} = O_{cur}$, and the optimal entropy $\mathcal{S}_{opt} = \mathcal{S}_{cur}$.

Step 2 Construct the neighborhood of O_{cur} and determine $CSS(O_{cur})$. Then randomly pick up candidates from $CSS(O_{cur})$ to construct L according to H.

Step 3 Refine the candidates in L and find O_{best}. If $\mathcal{S}_{cur} > \mathcal{S}(X, O_{best})$, update variables $O_{cur} = O_{best}, \mathcal{S}_{cur} = \mathcal{S}(X, O_{best})$.

Step 4 Increase the tabu-time of each element in the tabu list H, and remove the elements whose tabu-time is greater than the limitation.

Step 5 If the entropy is convergent, go to Step 6; otherwise, go to Step 2. In implementation, we assume that the entropy is convergent when it does not change after iterating 500 times.

Step 6 If $\mathcal{S}_{opt} > \mathcal{S}_{cur}$, update variables $O_{opt} = O_{cur}$ and $\mathcal{S}_{opt} = \mathcal{S}_{cur}$. If the aberrance-time is equal to zero, go to Step 7; otherwise, decrease the aberrance-time and perform the aberrance operation. Then set $\mathcal{S}_{cur} = \mathcal{S}(X, O_{cur})$, go to Step 2.

Step 7 Return O_{opt} and $\mathcal{S}_{opt}$.

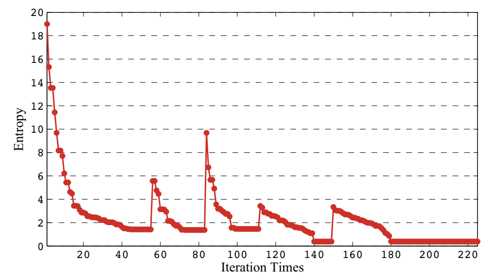


Figure 2: Entropy variation during iterations. A fluctuation means the entropy has met a convergence. There are totally 4 times of fluctuation since we assigned the limitation of the aberrance-time to be 4.

3.2 Examples

Figure 1 illustrates the procedure of finding the minimal entropy on a synthetic toy of 32 input data, which gives an intuition about how the active tabu search works. As the expectation in Equation (7) is to have low selection rate $p\%$ and high probability $\eta\%$, we take $p = 0.9$ and $\eta = 99$, causing $\sharp L = 512$. Then we take 4 to be the limitation of the tabu-time and the aberrance-time. Active tabu search finds the most ordered cycle as shown in Figure 1(c). Figure 2 shows the process of entropy minimization. A fluctuation in the figure means that an aberrance operation was performed. The entropy finally converged at 0.384314 after iterating approximate 180 times.

4 Fast Active Tabu Search

Since the calculation of the entropy of each element in the set L is independent, parallel computing is a good way to speed up the calculation. Fast active tabu search (FATS) is a parallel version of the active tabu search, which is implemented on the graphic processing unit (GPU) via the CUDA technology.

4.1 Implementation

As shown in Figure 3, the CUDA programming model has three layers, including *grid*, *block*, and *thread*. The single instruction set executed by each thread is called *kernel*. There is a small on-chip storage for fast data access, named *shared memory*. For more details, please refer to the document [NVIDIA, 2008].

FATS is implemented as follows. First, all transformations are represented as a unified format (o_i, o_j). Then, transformations involving swapping, shifting and inversing are generated for each datum of a cycle respectively. Therefore, there are totally three collections of transformations for the cycle. That is, L is composed of three collections. Finally, the design of FATS in CUDA is shown in Figures 4 and 5. One kernel is launched for a collection in each iteration. Inside a kernel, the number of the block depends on the scale of the

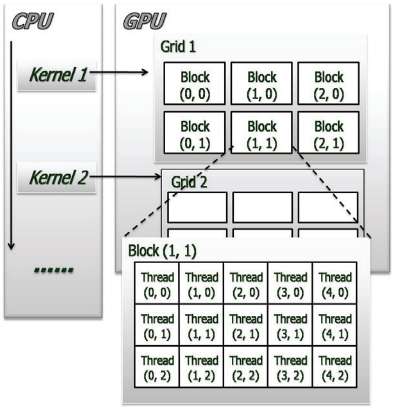


Figure 3: The programming model of CUDA.

input data ($\sharp D$). Each block processes one specific transformation for a datum of the cycle. If the manifold is embedded to an m-dimensional space, up to m threads will be created inside a block. Each thread only needs to deal with one specific dimensionality of vectors according to its ID. To calculate the entropy corresponding to the transformation (o_i, o_j), only the data fragments

$$(\ldots, o_{i-3}, o_{i-2}, o_{i-1}, \mathbf{o_i}, o_{i+1}, o_{i+2}, o_{i+3}, \ldots)$$

and

$$(\ldots, o_{j-3}, o_{j-2}, o_{j-1}, \mathbf{o_j}, o_{j+1}, o_{j+2}, o_{j+3}, \ldots)$$

need to be loaded to the shared memory of the i-th block of the specific kernel. The entropy of the order can be calculated through the variation of GEOMEN. After calculating the entropy for each transformation in a kernel, the minimum is extracted as the result and transmitted to CPU for succeeding iterations.

Given a data set X with a random order O, the procedure of FATS is described as follows:

Step 1 Set $O_{opt} = O$ and $\mathcal{S}_{opt} = \mathcal{S}(X, O)$. Copy the input data to GPU.

Step 2 Copy O_{opt} and $\mathcal{S}_{opt}$ to GPU. Then, generate swapping transformations (i.e. swapping collection) for each datum of the cycle according to the tabu list H. Transmit the swapping collection to the swapping kernel. After calculating the entropy of each transformation on GPU, transmit the minimal entropy $\mathcal{S}_{min}$ and its corresponding order O_{min} back to CPU. If $\mathcal{S}_{opt} > \mathcal{S}_{min}$, set $\mathcal{S}_{opt} = \mathcal{S}_{min}$ and $O_{opt} = O_{min}$.

Step 3 Execute the same operations to shifting and inversing respectively.

Step 4 Update the tabu list H.

Step 5 If the termination condition, which is the same as active tabu search, is satisfied, go to Step 6; otherwise go to Step 2.

Step 6 Return O_{opt} and $\mathcal{S}_{opt}$.

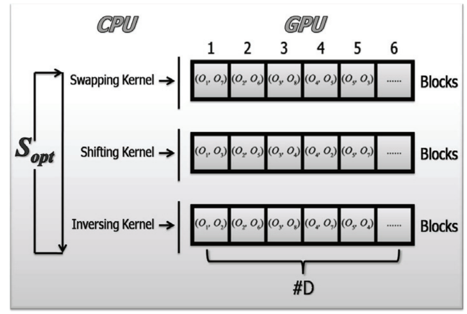


Figure 4: The block-level design of FATS. Each kind of kernel is implemented as a separate call to GPU. There are totally $\sharp D$ blocks of each kernel. The entropy $\mathcal{S}_{opt}$ is updated alternatively in the loop.

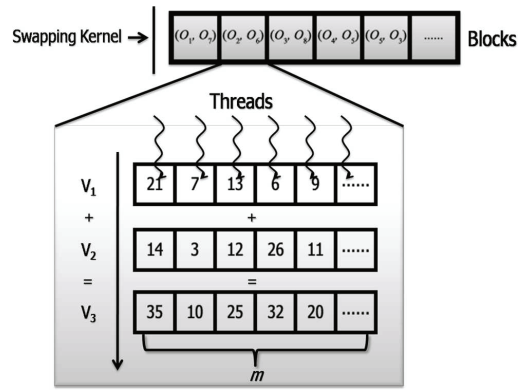


Figure 5: The thread-level design of FATS. For example, in the operation of addition, each thread only calculates one specific dimensionality of vectors.

4.2 Performance Analysis

In this work, we employ the NVIDIA GeForce 9800GT GPU, which is an instance of the CUDA architecture. FATS is tested on multiple data sets composed of 128-dimensional vectors. Table 1 shows the performance of FATS with respect to data scales. The running time is obtained after 500 iterations. From the table, we conclude that the performance of FATS is sensitive to the size of L ($\sharp L$), but robust to the data scale that is always the bottleneck of image retrieval. Thus, FATS is very suitable for image retrieval. Figure 6 shows the speedup ability of FATS. When the data scale increases, FATS even can reache higher speedup over active tabu search. This speedup stems from the efficiency of GPU computing cells and much exploitation of parallelism.

5 Application to Image Retrieval

This section presents the application of FATS to image retrieval. In our framework of image retrieval, we view an image as a point in the feature space, and compute the relevance of images from the corresponding optimal cycle through entropy minimization. For retrieval, we need to insert a query image into the obtained cycle. The inserting position means that inserting has the least effect on the GEOMEN value. That

Table 1: Running time(secs) of FATS

Scale \ $\sharp L$	$2^{10} \times 3$	$2^{11} \times 3$	$2^{12} \times 3$	$2^{13} \times 3$	$2^{14} \times 3$
256	2.109	3.813	7.219	14.063	26.859
512	2.360	3.984	7.203	13.734	26.547
1024	2.875	4.500	7.687	13.968	26.343
2048	3.738	5.563	8.781	15.016	27.125
4096	4.859	6.975	10.589	17.156	29.219
8192	5.880	8.371	12.495	21.215	33.156

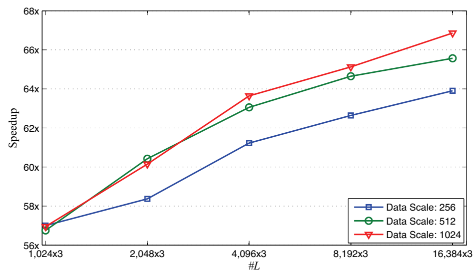


Figure 6: Speedup ability of FATS.

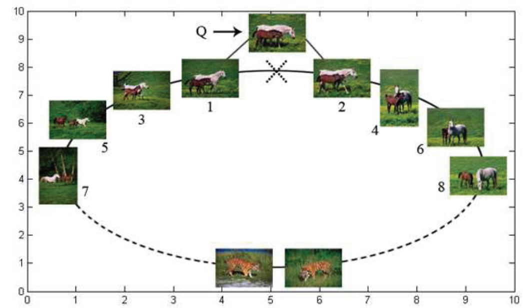


Figure 7: Illustration of the optimal cycle of images.

Table 2: Impact of $\sharp L$ on the performance of FATS

$\sharp L$	$p\%$	$\eta\%$	Time(secs)	Entropy
$2^{12} \times 3$	0.025	95.37	198	2,291.32
$2^{13} \times 3$	0.015	97.50	339	2,043.45
$2^{14} \times 3$	0.010	99.25	514	1,878.41
$2^{15} \times 3$	0.009	99.99	895	1,766.69
$2^{16} \times 3$	0.005	99.99	1,543	1,704.79

is, after inserting a new image at this position, the variation of the entropy ($\Delta\mathcal{S}$) is less than inserting at other positions. Considering that the inserting position probably locates at the boundary between image classes, we will alternately compute the ranking score on its local neighborhood. Since manifolds are locally flat, Euclidean distance is good enough to measure the ranking score.

Given an image database and a query image $\mathcal{Q}$, the proposed framework for image retrieval is briefly described as follows:

Step 1 Construct Equation (1) in the input image space.

Step 2 Solve the GEOMEN minimization problem using FATS, and find the optimal cycle with the order O_{opt}.

Step 3 Insert $\mathcal{Q}$ into O_{opt}. The measure of determining the inserting position is that $\Delta\mathcal{S}$ must be the least after inserting.

Step 4 Rank the nearest neighbors of $\mathcal{Q}$ along the cycle according to Euclidean distance between images. The less the distance, the more the relevance.

Step 5 Return the relevant images.

Figure 7 demonstrates the procedure of inserting and ranking for image retrieval. The query image $\mathcal{Q}$ was inserted into the position between images 1 and 2 where $\Delta\mathcal{S}$ was minimal after inserting. Then the relevant images are chosen from the neighbors of $\mathcal{Q}$ along the cycle and ranked as $1, 2, 3, \ldots$ according to Euclidean distance until enough images were returned.

6 Experimental Results

We performed several experiments to evaluate the effectiveness of FATS. Two data sets employed are respectively the Corel image data set and the 101 Object_Categories data set[1]. 1,024 images of 32 semantic categories (32 images for each) from the Corel data set were selected to build the first database. 6,272 images from Corel and another 1,920 images from 101 Object_Categories together build the second database (8,192 images in total). The second database was much more heterogeneous than the first one.

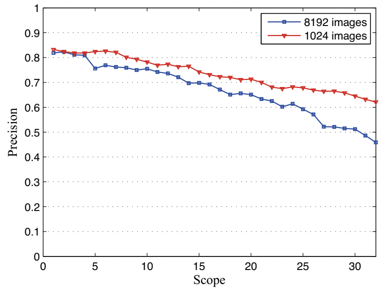
We used the combination of a 64-dimensional color histogram and a 64-dimensional color texture moment (CTM) [Yu *et al.*, 2002] to represent an image. The color histogram is calculated using $4 \times 4 \times 4$ bins in HSI color space. CTM gives a rough and robust texture characteristics, which utilizes Local Fourier Transform to extract features in each channel of the $(SV \cos H, SV \sin H, V)$ color space.

Table 2 shows the impact of $\sharp L$ on the performance of FATS, which was on the second image database. Obviously, the larger the $\sharp L$, the less the entropy, but the more the time. Thus, to balance the time and entropy, we fix $\sharp L$ to be $2^{15} \times 3$ in our experiments. The reason that $\sharp L$ is multiple of the power of 2 is that that kind of number leads to high performance and simple implementation on GPU.

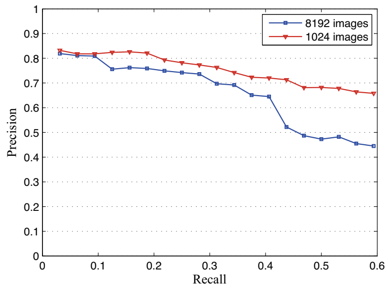
In order to exhibit the effectiveness of FATS, we compared the following two methods, *ridge regression* (RidegReg) [Cai *et al.*, 2007] and *support vector machine* (SVM) [Zhang *et al.*, 2001]. The *precision-scope* and *precision-recall* curves are used to evaluate the performance. In our context, the scope denotes the number of top returned images, and the precision is the ratio of the number of top relevant images to the scope. The recall represents the ratio of the number of retrieved relevant images to the total number of relevant images in an image database.

Figure 8 shows the average precision-scope and precision-recall curves of FATS on the two databases. The performance of FATS is not very sensitive to the size of image database although a little decrease of precision appears for the second

[1] http://vision.cs.princeton.edu/resources_links.html♯datasets

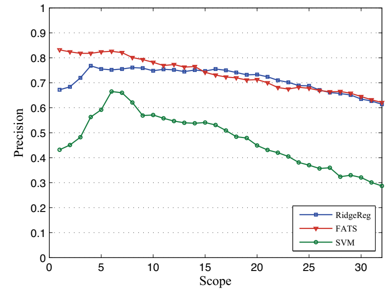


(a) x-axis: scope, y-axis: precision

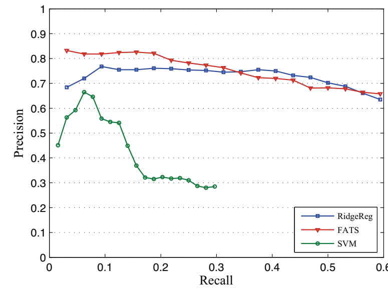


(b) x-axis: recall, y-axis: precision

Figure 8: Performance of FATS on two different image databases.



(a) x-axis: scope, y-axis: precision



(b) x-axis: recall, y-axis: precision

Figure 9: Comparison among FATS, RidgeReg and SVM.

image database. Figure 9 compares the performance of three methods on the first database. Obviously, our method has a better performance when the scope or the recall rate is less, and the performance is still comparable to the RidgeReg and remains superior to the SVM when the recall rate or scope becomes higher.

7 Conclusions

A novel framework is proposed for image retrieval in this paper. The retrieval is treated as searching for an ordered cycle in an image database. The optimal cycle can be found by minimizing GEOMEN through FATS. The use of GPU in the minimization yields a very considerable speedup. Our framework has an clear advantage over pervious manifold based methods: our method can directly rank and return relevant images and does not need to learn a mapping from the feature space to the unclear semantic manifold space, further avoiding the unnecessary exploration on the dimensionality of the semantic space.

References

[Cai *et al.*, 2007] D. Cai, X. He, and J. Han. Regularized regression on image manifold for retrieval. In *MIR '07: Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 11–20. New York, NY, USA, 2007. ACM.

[Ertekin *et al.*, 2007] S. Ertekin, J. Huang, and C.L. Giles. Active learning for class imbalance problem. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 823–824. New York, NY, USA, 2007. ACM.

[Glover, 1986] F. Glover. Future paths for integer programming and links to artificial intelligence. In *Computers and Operations Research*, volume 13, pages 533–549. 1986.

[He *et al.*, 2004] X. He, W. Ma, and H.-J. Zhang. Learning an image manifold for retrieval. In *MULTIMEDIA '04: Proceedings of the 12th annual ACM international conference on Multimedia*, pages 17–23. New York, NY, USA, 2004. ACM.

[Jing and Baluja, 2008] Y. Jing and S. Baluja. Visualrank: Applying pagerank to large-scale image search. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 30, pages 1877–1890. November 2008.

[NVIDIA, 2008] NVIDIA. Compute Unified Device Architecture: Programming Guide, version 2.1, Nov 2008.

[Yu and Tian, 2006] J. Yu and Q. Tian. Learning image manifolds by semantic subspace projection. In *MULTIMEDIA '06: Proceedings of the 14th annual ACM international conference on Multimedia*, pages 297–306. New York, NY, USA, 2006. ACM.

[Yu *et al.*, 2002] H. Yu, M. Li, H.-J. Zhang, and J. Feng. Color texture moments for content-based image retrieval. In *International Conference on Image Processing*, pages 24–28, 2002.

[Zhang *et al.*, 2001] L. Zhang, F. Lin, and B. Zhang. Support vector machine learning for image retrieval. In *Proceedings of IEEE international conference on Image Processing*, volume 2, 2001.