International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Application of Data Compactness in Image Mining

Yuqing Song*, Yaohui Li, Shaoqing Mo

School of Automotive and Transportation, Tianjin University of Technology and Education, Tianjin, China
* Corresponding author's Email: yqsong7@hotmail.com

Abstract: Image mining is concerned with knowledge discovery in image databases. With the advance of multimedia technology and growth of image collections, it is becoming crucial to analyze the compactness of image data and apply it to image mining. In this paper, we study the class compactness and boundary compactness of image data, which are used in image classification and data confining, respectively. The data confining procedure produces a relevance graph representing relevant image pairs and their relevancy. Based on relevant image pairs, a manifold learning technique is applied to compute distances between images and manifolds of images. Image retrieval is based on these distances. The effectiveness of the proposed approach has been validated by experiments on real-world images.

Keywords: Class compactness; Boundary compactness; Manifold learning; Image mining; Image classification; Image retrieval

1. Introduction

Progresses in the image acquisition and storage technology have led to tremendous growth in the significantly large and detailed image databases [1]. A recurring problem in computer vision and pattern recognition is knowledge discovery from image databases. Much more than just an extension of data mining to image domain, the image mining is an interdisciplinary endeavor to address this problem, which has gradually become the attention focus of research community. Image mining is a technique to extract patterns, implicit knowledge, and/or image data relationship which are not explicitly stored in images [2]. Applications include, but are not limited to, web data mining, image retrieval, medical and healthcare informatics, satellite image analysis, and mineral forecast.

The main intention of image mining is to generate considerable patterns without any information of the image content, the patterns types are different [3]. Frequently-used image mining techniques include: image similarity search, image association rule mining, image classification, image clustering, and neural networks. When manual image annotation becomes more and more unfeasible, image search, based on content similarity becomes popular. A lot of image retrieval systems adopt the similarity-based paradigm, including QBIC (IBM Query by Image Content) [4], VisualSEEk [5], Virage's VIR Image Engine [6], and Excalibur's Image RetrievalWare [7]. Image association rules provide information in image databases, such as interesting but non-obvious spatial or temporal causalities. Many association rule mining methods [8][9] have been proposed for image databases. The main objective of the image classification is to decide whether an image belongs to a certain category or not. Uehara et al. [10] used a binary Bayesian classifier to achieve a systematic image classification, where images are divided into two types: indoor and outdoor. Vailaya et al. [11] proposed a hierarchical classification of vacation images: at the highest level, images are classified as indoor or outdoor; outdoor images are further classified as city or landscape; finally, a subset of landscape images is classified into sunset, forest, and mountain classes. Image clustering methods partition images into clusters such that the images within the same cluster are similar to each other. Unlike the classification, the image clustering is unsupervised. Many image clustering methods [12][13] have been successfully used to better organize, represent, and browse images. A neural

network is a massively parallel distributed processor consists of several simple processing units, each has natural propensity of storing experiential knowledge and making knowledge available for use [14] and [15] presents noteworthy research work that applies neural network to image mining.

We distinguish two kinds of frameworks for characterizing image mining systems: function driven and information driven. Most existing image mining systems [16][17] are function driven. Multimedia Miner [16], a system developed by researchers from Simon Fraser University, is used as a framework belonging to the first category. The system constructs multimedia data cube facilitating multiple dimensional analyses of multimedia data, primarily based on visual content, and the mining of various kind of knowledge including summarization, comparison, classification, association, clustering. Function-driven architecture cannot effectively handle different levels of information representation in image mining. Zhang proposed an information driven framework for image mining [18]. The framework distinguishes four levels of information: the pixel level, the object level, the semantic concept level, and the pattern and knowledge level. A high dimensional indexing schemes and the retrieval techniques are also included to support the flow of information among the levels. This framework makes the first step towards capturing the different levels of information present in image data and addressing the question of what are the issues and challenges of discovering useful patterns/knowledge from each level.

The study of image mining is still in its infancy. With the advance of multimedia technology and growth of image collections, it is becoming crucial to analyze the compactness of image data and apply it to image mining. We can see the world, classify and analyze various scenes; in this process, data compactness plays an very important role. In the ever-changing world, we observe objects of different types; the appearance of objects in each type has a relatively stable and compact model. The stability and compactness in appearance indicate data compactness. In this paper we study the class compactness and boundary compactness of image data, which are used in image classification and data confining, respectively. The data confining produces a relevance graph representing relevant image pairs and their relevancy. Based on relevant image pairs, a manifold learning technique is applied in the computation of distances between images and manifolds of images. Image retrieval is based on these distances.

This paper is an extension of our previous work [19]. The rest of the paper is organized as follows. In the Section II, we will introduce a mathematical representation of image mining. Section III presents our approach for image classification by class compactness. Section IV proposes our scheme for data confining by boundary compactness. Image retrieval by manifold learning is elaborated in Section V. Section VI reports the experiments. Section VII concludes.

2. Mathematical Representation of Image Mining

Image mining is concerned with knowledge discovery in image databases. It is an effort to transform the low level image features into patterns, descriptions, and/or object relationship. Let R^d be the feature space of an image set,

$$X = \{x_1, x_2, ..., x_n\} \subset R^d$$

be the vector representation of the images in the feature space. The scenes of the images are divided into *m* categories:

$$\Psi = \{\psi_1, \psi_2, ..., \psi_m\}$$

Image clustering or classification can be represented as a mapping $g: X \longrightarrow \Psi$. In the real world, scenes in the same category are very different under different conditions, the images of these scenes form a manifold in the feature space. For large quantities of image data, we can use a manifold learning approach to investigate the problem of the semantic representation of images. Manifold Learning pursues goal to embed originally high dimensional data in a lower dimensional space, while preserving characteristic properties. For an image scene ψ_i, a manifold learning is a mapping $g_i: g^{-1}(\psi_i) \longrightarrow R^{di}$. Put together, the scene space can be formulated as,

$$Y = \bigcup_{i=1}^{m} \{\psi_i\} \otimes R^{di}$$

and the goal of image mining is to find a mapping from the image set *X* to the scene space *Y*, $h: X \longrightarrow Y$, where $h(x) = (g(x), g_i(x))$, and $g(x) = \psi_i$. Image association rule mining aims to find a set of association rules to reveal and represent the occurrence frequency of a group of objects/features, or their relationship. A typical association rule can be written as $P \longrightarrow Q[s\%, c\%]$, where *P* and *Q* are predicated, *s*% is the support of the rule, and *c*% is the confidence. It is common to use *P* for low level features and *Q* for semantic features so that we can use association rules to infer image semantics from low level features. Thus $p \subseteq R^d$, $Q \subseteq Y$, and every association rule is represented as a point in the

association with the rule space $\varphi(R^d) \otimes \varphi(Y)$, where $\phi(\bullet)$ represents the power set of a set.

The above mathematical representation of image mining can be translated into an image mining framework, shown in Figure 1.

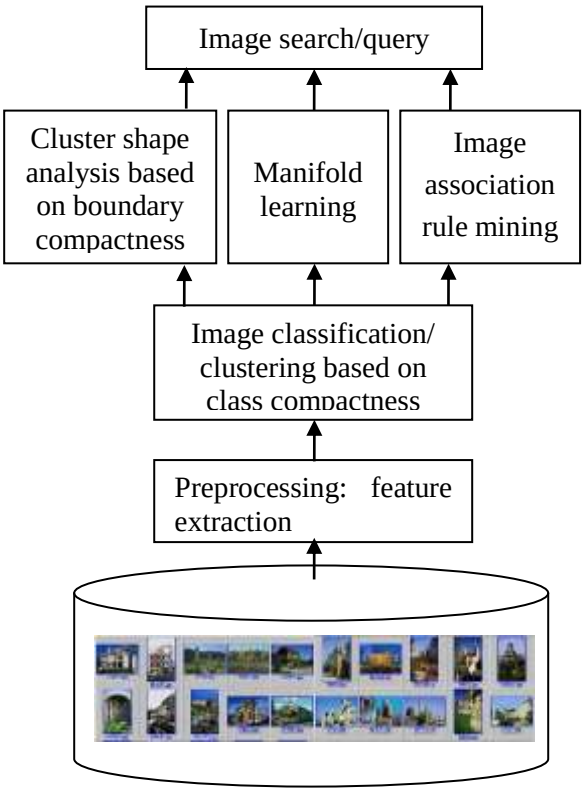
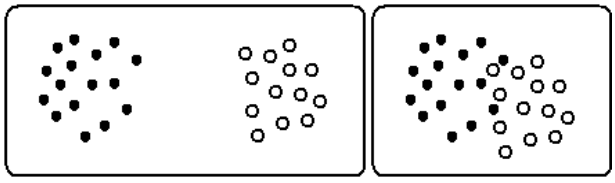


Figure 1 The framework of image mining

3. Image Classifying by Class Compactness

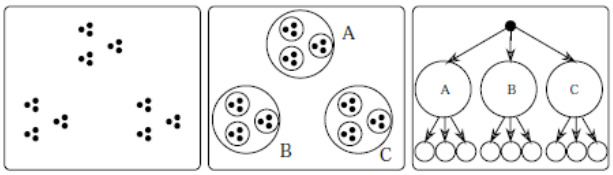
One main problem of image classification is that the relationship between inter-class distance and intra-class distance is not fully explored. To address this problem, we introduce an *iCluster Tree* model. An isolation cluster, or icluster, is a connected subset whose inter-subset distance to its complement (ECD, External Connecting Distance) is longer than its intra-subset distance (ICD, Internal Connecting Distance).



(a) (b)

Figure 2 (**a**) Two compact classes with long ECD and short ICD (**b**) Two classes which are not as compact

See Figure 2. The compactness of an icluster is defined as ECD/ICD. We can prove that all iclusters form a rooted tree, called the *iCluster Tree*. See Figure 3.



(a) (b) (c)

Figure 3 (**a**) A data set (**b**) The iclusters (**c**) The Icluster Tree

We apply this model to image classification. Let the training and test sets respectively be represented as $X_{tn} = \{x_1, x_2, \ldots, x_t\}$ and $X_{tt} = \{x_{t+1}, x_{t+2}, \ldots, x_n\}$. The predefined class set is $Y = \{Y_1, Y_2, \ldots, Y_m\}$. On the training set X_{tn}, there is a mapping: $f_{tn}: X_{tn} \rightarrow \{1, 2, \ldots, m\}$ which assigns each training point to a preset class. Supervised classification aims to extend the mapping f_{tn} to the test set X_{tt}.

We combine the two sets into one set:

$$X = X_{tn} \cup X_{tt} = \{x_1, x_2, \ldots, x_t, x_{t+1}, x_{t+2}, \ldots, x_n\}$$

and let G be a graph constructed on X and T be the icluster tree of G. Each icluster C has a *training histogram*, which is a m-tuple: $(|C \cap Y_1|, |C \cap Y_2|, \ldots, |C \cap Y_m|)$. The training histograms of iclusters can be computed recursively. Initially we compute the histogram of each leaf iclusters, i.e., an icluster with one point. Let the point be v and the histogram is :

$$(|\{v\} \cap Y_1|, |\{v\} \cap Y_2|, \ldots, |\{v\} \cap Y_m|)$$

For each non-leaf node, we add the histograms of all its children to get the histogram of this node. The *concentration ratio* of C is defined as:

$$\text{ConcentrationRatio}(C) = \frac{Max_{1 \le i \le m}(|C \cap Y_i|)}{|X_{nt}|}$$

An icluster C is called *concentration* on class Y_k if its concentration ratio is no less than a given threshold T_ratio and

$$|C \cap Y_k| = Max_{1 \le i \le m}(|C \cap Y_i|)$$

We make a top-down search to find all concentrated iclusters: we start with the root node; for each node, if it's concentrated (with respect to a given threshold) then add it to the result queue, otherwise repeat the process on its child nodes. The result queue has a property that the concentrated iclusters in the queue do not overlap. For a test point p in a concentrated icluster (concentrated on class Y_i) in the queue, we let $f(p) = i$, where f is the extended function of f_{tn}. However not all points appear in these concentrated iclusters. Let Z be the set of points classified thus far and suppose $Z = Z_1 \cup Z_2$

$\cup \ldots \cup Z_m$ such that $f(Z_i) = i$ for $1 \le i \le m$, i.e., points in each Z_i are assigned to the class Y_i. For any point q in $X-Z$, we calculate its distances to the centroids of $Z_1, Z_2, \ldots, Z_m$ respectively and add q to the set Z_j if its distance to the centroid of Z_j is the shortest. The whole data set X is finally classified as $Z_1, Z_2, \ldots, Z_m$, which are corresponding to the m preclasses, respectively.

The above discussion is formalized in following algorithms .Algorithm 1 aims computing the training histograms for all icluster nodes in an icluster tree t. It represents the training histograms with a function hF: $\{nd \mid nd \text{ is a node of } t\} \times \{1, 2, \ldots, m\} \rightarrow \mathbf{N}$, where $\mathbf{N}$ is the natural number set including 0. When the algorithm finishes, the histogram of a node nd is $(hF(nd,1), hF(nd,2), \ldots, hF(nd,m))$. Algorithm 2 takes the input as a node in an icluster tree, the histogram function, and a concentration threshold; it outputs a queue of concentrated iclusters in the subtree starting with the given node. Algorithm 3 calls Algorithm 1 to compute the histograms and then recursively calls the Algorithm 2 to find all concentrated iclusters in an icluster tree. Algorithm 4 constructs the icluster tree, calls Algorithm 3 to find concentrated iclusters, and then classifies the whole data set X into m classes $Z_1, Z_2, \ldots, Z_m$, which are corresponding to the m preclasses, respectively.

Algorithm 1 takes $\theta(n)$ space for the histogram function hF and computes the histograms for all nodes in a reverse depth-first order. So the running time is linear to the size of the icluster tree, which is $\theta(n)$. The time and space needed by Algorithm 1 are both $\theta(n)$. Algorithm 3 takes $O(n)$ space to represent the queue of concentrated iclusters and recursively find the concentrated iclusters. The time and space of Algorithm 3 are both $O(n)$. For Algorithm 4, after the graph and its icluster tree are created, it calls Algorithm 3, taking $\theta(n)$ time and space; creates and initializes the classification function f, taking $\theta(n)$ time and space; extends the function f to the test points in the concentrated iclusters, taking $O(n)$ time and space; and then classifies the remaining points, taking $O(n)$ time and space. So the overall time and space for the data classification in Algorithm 4 are both $\theta(n)$.

Algorithm 1. *Compute_TrainingHistograms*(t)
Input: an icluster tree t
Output: the training histograms of all nodes
Begin
(1) $hF \leftarrow$ a function from $\{nd \mid nd \text{ is a node of } t\} \times \{1, 2, \ldots, m\}$ to $\mathbf{N}$
(2) hF is initialized such that $HF(nd, i) = 0$ for all (nd, i).
(3) **For** each node nd of the tree t **do**:
a. **If** nd is a leaf node, containing a point $v \in Y_i$ for some i in $\{1, 2, \ldots, m\}$ **then do** $hF(nd, i) \leftarrow 1$.
b. **If** nd is a nonleaf node **then for** each its child node nd_c and each i in $\{1, 2, \ldots, m\}$ **do** $hF(nd, i) \leftarrow hF(nd, i) + hF(nd_c, i)$.
(4) return hF.
End

Algorithm 2. *Compute_ConcentratediClusters*(nd, hF, T_ratio)
Input: an icluster node nd, a histogram function hF, and a concentration threshold T_ratio
Output: a queue of concentrated iclusters in the subtree starting with the given node nd
Begin
(1)$Q \leftarrow$ an empty queue
(2) If nd is concentrated with respect to T_ratio:
a. append nd to Q
b. rerutn Q
(3)For each child nd_c of nd do:
b. append ConcentratediClusters(nd_c, T_ratio) to Q
(4)return Q;
End

Algorithm 3 *Compute_ConcentratediClusters*(t, T_ratio)
Input: an icluster tree t and a concentration threshold T_ratio
Output: a queue of concentrated iclusters
Begin
(1)$hF \leftarrow$ *Compute_TrainingHistograms* (t); //Calling Algorithm 1
(2)$nd \leftarrow$ root of *tree*
(3)return *Compute_ConcentratediClusters*(nd, hF, T_ratio); //Calling Algorithm 2
End

Algorithm 4. *DataClassify* (X_{tt}, X_{tn}, f_{tn}, T_ratio)
Input: a test set $X_{tt} = \{x_{t+1}, x_{t+2}, \ldots, x_n\}$,
a training set $X_{tn} = \{x_1, x_2, \ldots, x_t\}$,
a classifcation f_{tn} on X_{tn},
and a concentration threshold T_ratio
Output: a classifcation f on $X = X_{tn} \cup X_{tt}$
Begin
(1)Construct a neighborhood graph G on $X = X_{tn} \cup X_{tt}$
(2)$t \leftarrow$ the *iClusterTree* of G
(3)$Q \leftarrow$ *Compute_ConcentratediClusters*(t, T_ratio) //Calling Algorithm 3
(4)$f \leftarrow$ a function from X to $\{1, 2, \ldots, m\}$
(5)f is initialized such that $f(x) = f_{tn}(x)$ for all training points x in X_{tn}.
(6)**For** each nd in Q which is concentrated on some Y_k **do**
a. **For** each test point x in nd **do** $f(x) = k$
(7)Let Z be the set of points already classified thus far and suppose $Z = Z_1 \cup Z_2 \cup \ldots \cup Z_m$ such that:

$f(Z_i) = i$ for $1 \le i \le m$

(8)**For** each point q in $X - Z$ **do**

b. Calculate its distances to the centroids of Z_1, Z_2, … , Z_m respectively

c. Find Z_j such that q's distance to the centroid of Z_j is the shortest

d. $f(q) \leftarrow j$

e. Add q to the set Z_j

(9)Return f.

End

4. Data Confining by Boundary Compactness

In real applications, high-dimensional image data is difficult to interpret as it requires more dimensions to represent. As a dimension reduction method, manifold learning provides an explicit representation for the useful implicit information hidden in the original feature space. But the internal topological and the differential structure has been disappeared in the dimensionality reduction process. On the other hand, existing surface reconstruction methods only work for low-dimensional, mostly 2D or 3D, data. We introduce boundary compactness to study the shape of the data set in the original high-dimensional feature space. The relationship between the data classification and data confining is illustrated by Figure 4.

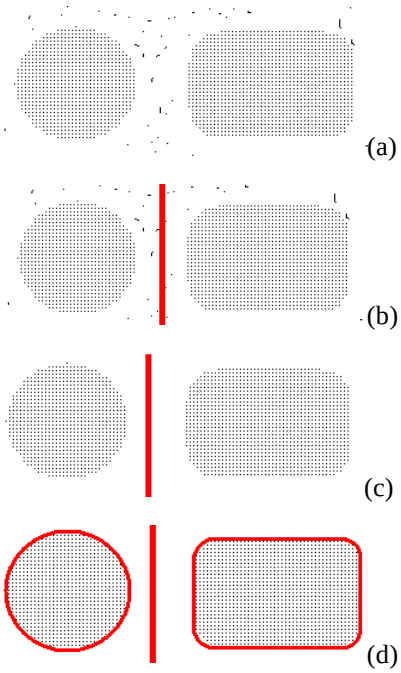


Figure 4 (**a**) A data set (**b**) Two classes of the data set (**c**) Noise removal (**d**) The boundaries of the classes

Given a K-dimensional set of n data points, to construct the Delaunay diagram, we discuss two cases: (1) $K \le 2$ and (2) $K > 2$. When $K \le 2$, there exist O(n log n) algorithms (that is optimal) to compute the Voronoi diagram and Delaunay triangulation [20]. In dimension $K > 2$, Delaunay triangulations can be computed in $O(n^{\lceil K/2 \rceil})$ time [21]. For the boundary fitting-by-erosion process, the graph to be "eroded" depends on K. When $K \le 3$, we choose Delaunay triangulation; when $K > 3$, the complete graph is used. In each case we call the graph to be eroded the *fat graph*.

For 3D and higher dimensional data, the erosion process is based on local density and controlled by boundary compactness. When the erosion process stops, we get the data shape (Figure e). Given an n-dimensional ($n > 2$) data set X and some x in X, we select m points closest to x in the MST, and compute the average length $\delta(x)$ of the MST edges connecting the m points and x. $\delta(x)$ indicates the local density at x. For any edge pq in the fat graph, we define its *boundary compactness* as formula:

$$\frac{|P - Q|}{(\delta(P) + \delta(Q))/2}$$

We remove from the fat graph the edges of boundary compactness greater than a threshold and get a graph, called the *relevance graph*. In the relevance graph, points connected by an edge represent closely related instances in real world.

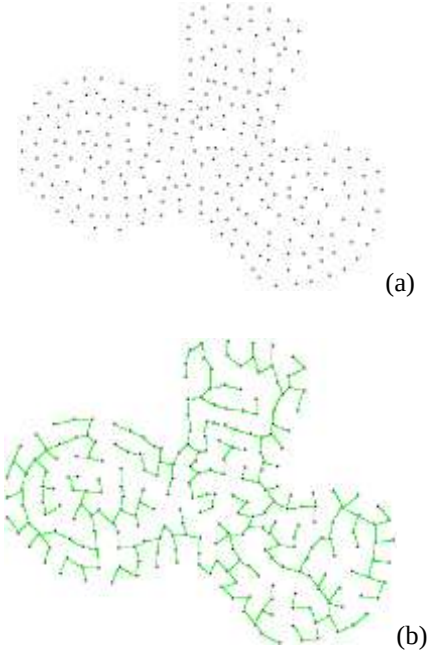


Figure 5 Data confining by cutting the Delaunay triangulation at "good" boundary gaps: **(a)** the data set **(b)** its MST

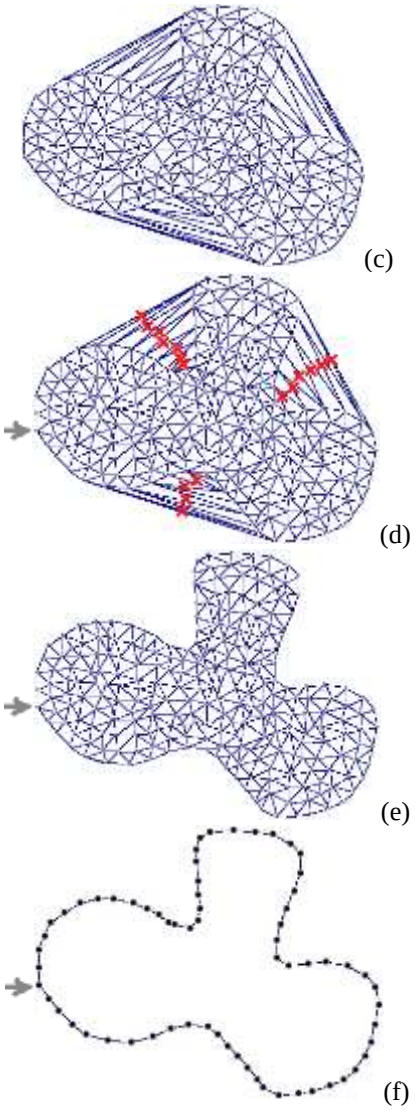


Figure 5 Data confining by cutting the Delaunay triangulation at "good" boundary gaps: (**c**) its Delaunay triangulation (**d**) removal of boundary gaps of big boundary compactness (**e**) the shape of the data (**f**) the boundary

We now introduce the algorithm (Algorithm 5) to compute $\delta(x)$ for each vertex x in the MST. In the preprocessing step, for each vertex x in MST, we sort its neighbors by their distances to x. In Algorithm 5, for each x in MST, we consider MST as a tree rooted at x, and search the rooted tree to find m closest neighbors of x and compute $\delta(x)$.

Algorithm 5. Delta (MST, x, m)
Input: MST, a vertex x in MST, and an integer m
Output: $\delta(x)$
Begin
1. $\delta(x) \leftarrow 0$
2. $j \leftarrow$ the number of neighbors of x in MST
3. $i \leftarrow \min(m, j)$
4. Take first i closest neighbors of x, save them to a list L
5. **For** each v in L, parent(v) $\leftarrow x$
6. $k \leftarrow 0$
7. **while** $k < m$
a) $a \leftarrow$ the first item of L
b) $d(a) \leftarrow$ the distance between a and parent(a)
c) $\delta(x) \leftarrow \delta(x) + d(a)$
d) $k \leftarrow k + 1$
e) $j \leftarrow$ the number of neighbors of a in MST
f) Take first $\min(m–k, j–1)$ closest neighbors of a excluding parent(a), save them to a list L_2
g) For each v in L_2, parent(v) $\leftarrow a$
h) $i \leftarrow \min(m–k, |L_2|+|L|)$
i) Merge sort L_2 and L, take the first i vertices and save to L
8. $\delta(x) \leftarrow \delta(x) \div m$
9. **return** $\delta(x)$
End

We make a straightforward analysis for the space and time. The δ (•) function takes O(n) space. So is the parent (•) function. The list L uses O(m) space. The total space is O(n), including the MST representation. As for time, we use $O(n^2)$ time to compute the MST. In the preprocessing, at an x of j neighbors in MST, O(j log j) time is needed to sort its neighbors. So the preprocessing uses O(n log n) time. In Algorithm 5, for each x, Steps 1-6 takes O(m) time. Steps 7(a-e) takes O(1) time; Steps 7(f-g) takes O($m–k$) time; Step 7(h-i) takes O($m–k$) time. So the time for Step 7 is $O(m^2)$. The time needed by Algorithm 5 is $O(m^2)$. It takes $O(n \log n + n\, m^2) = O(n \log n)$ time to run the preprocessing step and Algorithm 5 on all x in X. The total time including the MST creation is $O(n^2)$.

Once the relevance graph is created, it can be used for surface reconstruction and investigation of the topological and differential structure of a data set. In this paper the data confining process results in relevance graph for each data class, which is then used for image retrieval.

5. Image Retrieval by Manifold Learning

An image retrieval system is a computer system for browsing and retrieving images from a large image base. In this paper we will apply manifold learning techniques on the image retrieval. Many machine learning systems tend to be very slow when operating on high-dimensional data, as is known as the curse of dimensionality. In many applications, the observed data are found to lie on the low dimensional manifold embedded in the higher dimensional space. Manifold Learning, also referred to as non-linear dimensionality reduction, is a

technique to find the intrinsic structure of high dimensional data by mapping them to a lower dimensional manifold, while preserve characteristic properties.

A variety of manifold learning methods can be found in the literature. Topologically Constrained Isometric Embedding [22] uses both local and global distances to learn the intrinsic geometry of flat manifolds with boundaries. The algorithm filters out potentially problematic distances between distant feature points based on these properties of the geodesics connecting those points and their relative distance to the boundary of the feature manifold, thus avoiding an inherent limitation of the Isomap algorithm. RankVisu [23] is a mapping method designed to the preservation of neighborhood ranks rather than their dissimilarities. A mapping of data is obtained in which neighborhood ranks are as close as possible according to the original space.

In our approach, image retrieval is implemented by computing distances between points, between a point and a manifold, and between manifolds. See Figure s. Using the scheme introduced in [24], a manifold M is represented as a set of subspaces $M = \{C_1, C_2, \ldots, C_m\}$. We define the following distances:

(1) $d(x_1, x_2) = \|x_1 - x_2\|$, where x_1 and x_2 are two points, and $\|x_1 - x_2\|$ is the distance in the relevance graph.

(2) $d(x, C) = \min_{y \in C} \|x - y\| = \|x - x'\|$, where C is a subspace, and x' is the projection of the point x on C.

(3) $d(x, M) = \min_{C_i \in M} d(x, C_i) = \min_{C_i \in M} \min_{y \in C_i} \|x - y\| = \|x - x''\|$, where M is a manifold, and x" is the projection of the point x on M.

(4) $d(C, M) = \min_{C_i \in M} d(C, C_i)$.

(5) $d(M_1, M_2) = \min_{C_i \in M_1} d(C_i, M_2) = \min_{C_i \in M_1} \min_{C_j \in M_2} d(C_i, C_j)$, where M_1 and M_2 are two manifolds.

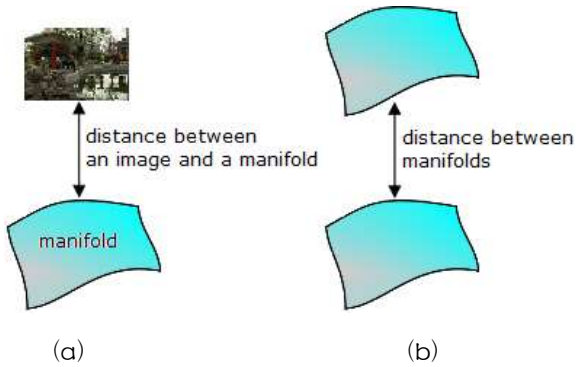


Figure 6 (**a**) The distance between an image and a manifold (**b**) the distance between manifolds

6. Experimental Validation

We first conducted experiments to validate the data confining algorithm, which was implemented in C++ and executed on a Dell OptiPlex GX270 PC with 2.80GHz Pentium 4 CPU and 1GB RAM. We tested on three 3D data sets, as is shown in Figure 7. The experiment statistics is listed in Table 1.

The relevance graphs of the 3D data sets are given in Figure 7 which well demonstrates the effecttiveness of our data confining algorithm.

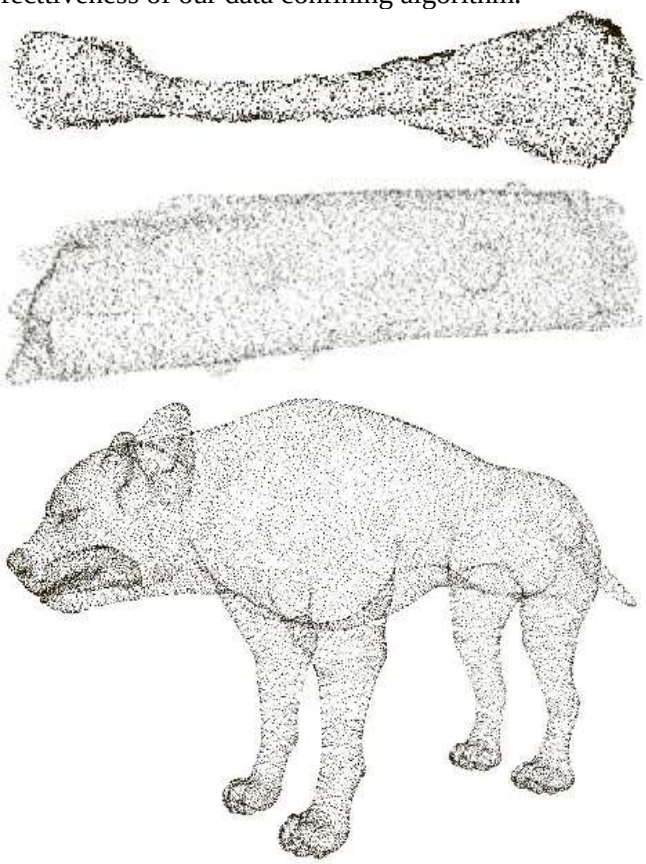


Figure 7 The three 3D data sets used in our experiments

Table 1 The experiment statistics of the three 3D data sets

No.	Shape	*n*	*e*	*tr*	*te*	*t*
1	bone	106,432	702,452	1505,539	738,466	1425
2	trunk	151,288	922,856	1945,543	950,397	2438
3	dog	290,115	1966,266	3456,477	1634,913	4974

For each data set, we list the number of points (*n*), the number of edges (*e*) in the relevance graph, the number of triangles (*tr*), the number of tetrahedra (*te*), and the computation time (*t*) in seconds.

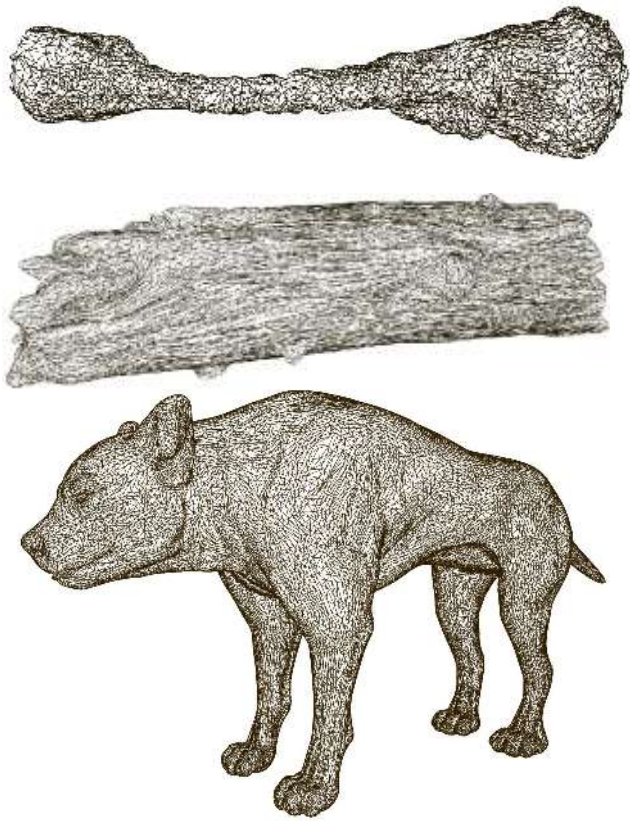


Figure 8 The relevance graphs of the three 3D data sets

We now report the experiments designed to test the effectiveness of our image mining approach. We used a web spider to collect 2000 images in the TUTE web site (www.tute.edu.cn). A subset of 200 images was randomly selected as the training set. The training images were manually classified into the following 10 classes: Opening Ceremony (开学典礼), Military Training (军训), Joining the party (入党), the Red Song Contest (红歌大赛), School-enterprise cooperation(校企合作), Lecture (讲座), International Exchange (国际交流), Equipment (设备), Classroom(教室), and Campus Scenery(校园风光). We then applied the proposed classification algorithm to classify the 2000 images, and used the data confining algorithm to compute the relevance graph. The manifolds of the image classes were constructed.

In the experiments, given an query image q, we search a best match image by finding the image m in the image base with the smallest $d(q, m)$. Given a set of query images, we first construct a manifold M_1, We search a best match image class by finding the image manifold M_2 in the image base with the smallest $d(M_1, M_2)$. The overall accuracy of the best matched image testing is 83%, and overall accuracy of the best match class testing is 87%. See Figure 9 for the sample images

Opening Ceremony

Military Training

Joining the Party

Red Song Contest

School-Enterprise Cooperation

Figure 9 Sample images

Lecture

International Exchange

Equipment

Classroom

Campus Scenery

Figure 9 Sample images

7. Conclusion

In this paper we investigate class compactness and boundary compactness of image data. Manifold learning techniques are applied in the computation of distances between images and manifolds of images. The introduced techniques have been tested by experiments.

Acknowledgments

This research was supported by the Natural Science Foundation of China under contracts No.61070112 and No. 61070116, and Hi-Tech Research and Development Program of China (863 Program) under contract No. 2009AA01Z317.

References

[1] O. R. Zaiane, J. Han, Z. Li, and J. Hou, Mining MultiMedia Data. In Proceedings of the 1998 conference of the Centre for Advanced Studies on Collaborative Research, pp 83-96, Toronto, Canada, November 1998.

[2] P. Stanchev, Using Image Mining for Image Retrieval. In the IASTED International Conference on Computer Science and Technology, pp 214-218, Cancun, Mexico, May 2003.

[3] R. Sudhir, A Survey on Image Mining Techniques: Theory and Applications. In Computer Engineering and Intelligent Systems, 2(6), 2011.

[4] QBIC™-IBM's Query By Image Content. http://wwwqbic.almaden.ibm.com.

[5] J. R. Smith and S.-F. Chang, VisualSEEk: a fully automated content-based image query system. In Proceedings of ACM Multimedia, pp. 87-98, 1996.

[6] J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C. Shu, The Virage Image Search Engine: An open framework for image management. In Proceedings of SPIE, Storage and Retrieval for Still Image and Video Databases, 1996.

[7] J. Dowe, Content-based retrieval in multimedia imaging. In Proceedings of SPIE. Storage and Retrieval for Image and Video Database, 1993.

[8] Q. Ding, Q. Ding, and W. Perrizo, Association rule mining on remotely sensed images using p-trees. In Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, pp 66-79 , May 2002.

[9] A. Lee, R. Hong, W. Ko, and Y. Liu, Mining Spatial Association Rules in Image Databases, In Information Sciences, 177(7): 1593-1608, Apr. 2007.

[10] Y. Uehara, S. Endo, S. Shiitani, D. Masumoto, and S. Nagata, A Computer-Aided Visual Exploration System for Knowledge Discovery from Images. In Proceedings of the Second International Workshop on Multimedia Data Mining, pp.102-109, San Francisco, CA, USA, August, 2001.

[11] A. Vailaya, M. Figueiredo, A. Jain, and H. Zhang, Image Classification for Content-Based Indexing. IEEE Transactions on Image Processing 10 (1) : 117 -130, 2001.

[12] M. Maheshwari, S. Silakari, M. Motwani, Image Clustering Using Color and Texture. In First International Conference on Computational Intelligence, Communication Systems and Networks, pp 403-408, July 2009.
[13] Y. Yang, D. Xu, F. Nie, S. Yan, and Y. Zhuang, Image Clustering using Local Discriminant Models and Global Integration. In IEEE Transactions on Image Processing, 19(10): 2761-2773, Oct. 2010.
[14] S. Haykin, Neural Networks: a comprehensive foundation, Prentice Hall, 1999.
[15] M. Antonie, O.R. Za ̈ane, and A. Coman, Application of Data Mining Techniques for Medical Image. In Proceedings of the Second International Workshop on Multimedia Data Mining, pp 94-101, San Francisco, CA, USA, August 2001.
[16] O. Zaiane, J. Han, Z. Li, S. Chee, and J. Chiang, MuhiMediaMiner: A System Prototype for MultiMedia Data Mining. In Proceedings of the International Conference on Management of Data - SIGMOD, 27(2): 581-583, 1998.
[17] M. Datcu and K. Seidel. Image information mining: exploration of image content in large archives. In Proceedings of the IEEE Conference on Aerospace, Vol.3, pp 253-264, 2000.
[18] J. Zhang, W. Hsu, M. Lee, An Information-driven Framework for Image Mining. In 12th Int. Conference on Database and Expert Systems Applications, pp 232-242, 2001.
[19] Y. Song, Y. Li, Image Mining by Data Compactness and Manifold Learning. In Proceedings of the 5th International Conference on Intelligent Networks and Intelligent Systems, pp 29-32, Nov. 2012.
[20] M. Shamos and D. Hoey, Closest-point problems. In Proceedings of the 16th IEEE Symposium on Foundations of Computer Science, pp 151-162, 1975.
[21] K. Clarkson and P. Shor, Applications of random sampling in computational geometry II. In Discrete & Computational Geometry, 4:387-421, 1989.
[22] G. Rosman, M. Bronstein, A. Bronstein, and R. Kimmel, Nonlinear Dimensionality Reduction by Topologically Constrained Isometric Embedding. In International Journal of Computer Vision, 89(1):56-68, 2010.
[23] S. Lespinatsa, B. Fertilb, P. Villemaina, J. Hérault, RankVisu: Mapping from neighbourhood network, in Neurocomputing, 72 (13–15):2964-2978, 2009.
[24] S. Roweis and L. Saul, Nonlinear Dimensionality Reduction by Locally Linear Embedding. In *Science*, 290(5500):2323–2326, December 2000.