Similarity Retrieval Based on SOM-Based R*-Tree

K.H. Choi[1], M.H. Shin[1], S.H. Bae[1], C.H. Kwon[2], and I.H. Ra[3]

[1] 375, Seosuk-Dong, Dong-Gu Kwangju , Computer Science & Statistics, Chosun University, Korea, 501-759
ckhplc@hanmir.com, minandjih@hanmail.net, shbae@chosun.ac.kr
[2] 604-5, Dangjung-Dong,Gunpo-si,Kyunggi-do, Division of IT, Computer Engineering , Hansei University, Korea, 435-742
kwmch@hotmail.com
[3] Depts. Electronic and Information Engineering, Kunsan National, Korea, 573-701

Abstract. Feature-based similarity retrieval has become an important research issue in multimedia database systems. The features of multimedia data are usually high-dimensional data. The performance of conventional multi-dimensional data structures tends to deteriorate as the number of dimensions of feature vectors increases. In this paper, we propose a SOM-based R*-tree(SBR-Tree) as a new indexing method for high-dimensional feature vectors. The SBR-Ttree combines SOM and R*-tree to achieve search performance more scalable to high dimensionalities. When we build an R*-tree, we use codebook vectors of topological feature map which eliminates the empty nodes that cause unnecessary disk access and degrade retrieval performance. We experimentally compare the retrieval time cost of a SBR - Tree with that of an SOM and an R*-tree using color feature vectors extracted from 40,000 images. The result show that the SOM-based R*-tree outperforms both the SOM and R*-tree due to the reduction of the number of nodes required to build R*-tree and retrieval time cost.

1 Introduction

With the increasing use of new database applications for dealing with highly multidimensional data sets, technology to support effective query processing with such a data set is considered an important research area. Such applications include multimedia databases, medical databases, scientific databases, time-series matching, and data analysis/data mining. For example, in the case of image searches, a typical query of content-based image retrieval [1] is "find images with similar colors, texture, or shapes in a collection of color images". The features used in this query are useful to discriminate multimedia objects (e.g., documents, images, video, music score etc.). A feature vector is a vector that contains a set of features, and usually hold high-dimensional data. Many indexing techniques [2][3][4][5] have been proposed to access such high-dimensional feature vectors effectively. These index trees wok effectively in low to medium dimensionality space (up to 20-30 dimensions). However, even a simple sequential scan performs better at higher dimensionalities[5].

This paper is organized as follows : In section 2, we provide an overview of related work. In Section 3, we present the algorithm of the SOM and R*-tree, and describe

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3038, pp. 234–241, 2004.

the SOM-based R*-tree proposed in this research. We experiment in order to compare the SOM-based R*-tree with the SOM and -tree alone in terms of retrieval time cost using color feature vectors extracted from 40,000 image. The experimental results are discussed in Section 4, and Section 5 presents the concluding remarks.

2 Related Works

In this Section, we describe the related work on clustering methods and high-dimensional index structures.

Clustering: There are supervised and unsupervised clustering methods for clustering similar data. During the training phase in supervised clustering, both the input data and the desired output are presented to the neural network. If the output of neural network differs from the desired output, the internal weights of the neural network are adjusted. In unsupervised clustering, the neural network is given only the input vectors and the neural network is used to create abstractions in the input space.
SOM is an unsupervised self-organizing neural network that is widely used to visualize and interpret large high-dimensional data sets. In this study, the reasons for using SOM are as follows: (i) No prior assumption is needed for distribution of data, (ii) the learning algorithm is simple, (iii) we do not need an external supervised signal for input and it learns self-organizationally, and (iv) similarity can be found automatically from multidimensional feature vector, and similar feature vectors are mapped onto neighboring regions on the topological feature map, in particular, highly similar feature vectors are mapped on the same node.

High-dimensional index structure: Many techniques, including R-tree, R^+-tree, SS-tree, SR-tree, X-tree, TV-tree, and Hybrid tree have been proposed to index feature vectors for similarity search. Despite various attempts at accessing high-dimensional feature vectors effectively, the current solutions are far from satisfactory. Although these index structures can scale to medium dimensionalities, above a certain dimensionality they are outperformed by a simple sequential scan through the database. This occurs because the data space becomes sparse at high dimensionalities, causing the bounding regions to become large[7]. We selected the R*-tree among other index techniques for the following reasons: (i) This index structure is the most successful variant of the R-tree, (ii) it can be applied to spatial data such as geography and CAD data, and (iii) it can be used as an index structure for feature space such as image retrieval currently.

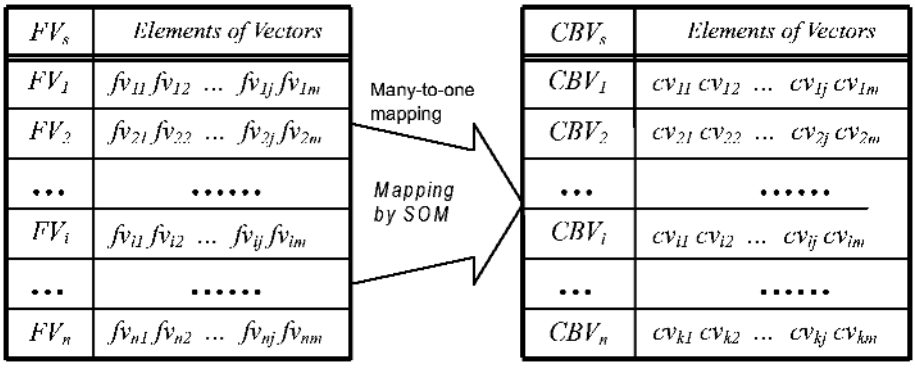
3 SOM-Based R*-Tree

The construction of a SOM-based R*-tree consists of two processes; clustering similar images and construction of R*-tree, as following.

Clustering similar images: We first generate the topological feature map using the SOM. We generate the BMIL by computing the distances between the feature vectors and codebook vectors from the topological feature map. The Best-Match Nodes(BMN) and its topological neighbors are moved closer to the input feature vector. As a result of learning ; the vector, which is generated on each node of the map, is called a codebook vector, and is represent by

$$CBV_i = [cv_{i1}, cv_{i2}, ..., cv_{ij}, ..., cv_{im}]^T,$$

where i($1 \le i \le k$) is the node number of the map, m is the number of input nodes, i. e., the dimensionality of the feature vector, and k is the number of map nodes.



(a) Feature vectors extracted from images (b) Codebook vectors generated by SOM

Fig. 1. Relationship between feature vector and codebook vector

Using the topological feature map, we classify similar image to the nearest node, which has the minimum distance between a given feature vector and all codebook vectors. This classified similar image of each node is called the best-matching-image-list (BMIL). Similarity between feature vectors and codebook vectors is calculated by the Euclidean distance. Best-match-node BMN_i is

$$BMN_i = \min_i \{ \| FV - CBV_i \| \}$$

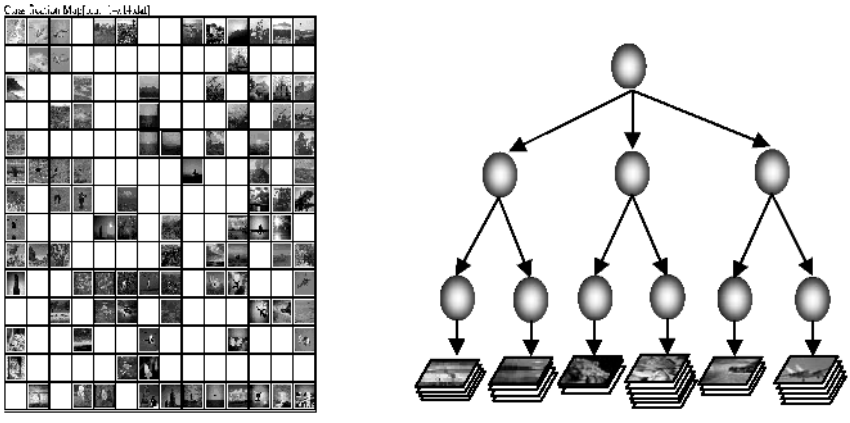
where FV is a feature vector. The relationship between feature vectors and codebook vectors is shown in Figure 1. Between these two kinds of vectors, there are many-to-one relationships based on the similarity between each feature vector. This means that empty nodes occur in a topological feature map when the BMIL is generated. Empty nodes refer to the portion of the node (vector) spaces that contains no matching feature vectors. Empty mode indexing causes unnecessary disk access, thus degrading search performance. The space requirement can be reduced by indexing only live nodes (in contrast to empty nodes).

Construction of R*-tree: In the construction of a SBR-Tree, we use the R*-tree algorithm[5]. Here, let one point on the n-dimensional space correspond to each

codebook vector of the topological feature map and the space covering all codebook vectors corresponds to the root node. In order to construct the R*-tree, we select a codebook vector from the topological feature map an entry. If it is an empty node, we select the next codebook vector. Otherwise, we determine the leaf node which insert codebook vector. The insertion algorithm of the R*-tree determines the most suitable subtree to accommodate the new entry(i. e., codebook vector) by choosing a subtree whose centroid is the nearest to the new entry. When a node or a leaf is full, the R*-tree reinsets or splits its entries. Otherwise, the new entry is add in the node or leaf. A leaf of the SOM-based R*-tree has the following structure:

$$L : (E_1, ..., E_i, ..., E_p) \quad (m \le p \le M)$$
$$E_i : (OID, \mu)$$

A leaf L consists of entries $E_1, ..., E_i, ..., E_p$ (m≤p≤M), where m and M are the minimum and the maximum number of entries in a leaf. Each entry contains an OID and its MBR μ. The node structure of the SBR-Tree is same as that of the R*-tree as shown in Fig. 2.



(a) example of topological feature map (b) SOM-based R*-tree structure using topological feature map eliminated empty nodes

Fig. 2. SOM-based R*-tree structure

4 Experiments

We performed experiments to compare the SOM-based R*-tree with a normal SOM and R*-tree. Our image database contains still color images. The experimental image database currently consists of 40,000 artificial/natural images, including landscapes, animals, buildings, people, plants, CG, etc., from H^2soft and Stanford University. We fixed the image size at 128×128 pixels. All experiments were performed on a COMPAQ DESKPRO(OS:FreeBSD 3.4-STABLE) with 128 Mbytes of memory and all data was stored on its local disk.

4.1 Experimental Methodology

Feature Extraction: In this study, we extract practice[1]. One disadvantage of using Haar wavelets is that the computation tends to produce blocky image artifacts in the most important subbands. However, this drawback does not noticeably affect similarity retrieval[13]. The color space used in this paper for feature vectors is the TIQ-space (NTSC transmission primaries)[14] with luminance and color features from the image data, and use it in the experiments. To computer feature vectors, we use Haar wavelets[12], which are a kind of wavelet transform. Haar wavelets provide the fastest computations and have been found to perform well in practice[1]. One disadvantage of using Haar wavelets is that the computation tends to produce blocky image artifacts in the most important subbands. However, this drawback does not noticeably affect similarity retrieval[13]. The color space used in this paper for feature vectors is the TIQ-space (NTSC transmission primaries)[14] with luminance and chrominance information. We computed 5 level two-dimensional wavelet transforms for each of the three color spaces using Haar wavelets. Extracting the lowest submatrix for the color feature, we generated this submatrix as part of the feature vector. Each element of this feature vector represents an average of 32×32pixels of the original image. The color feature vector has 48 dimensions (=4×4×3,where 3 is the three channels of YIQ space).

Construction of SOM-based R*-tree. As shown in Table 1, we determine that the map size is almost the same as the number of images in the image databases. We generated the topological feature map using color feature vectors via the learning of the SOM, and the BMIL is generated using this feature map. The empty nodes occupied 53% to 60% of the original map size. As the map size becomes larger, the number of empty nodes increases. This means that images of high similarity are classified in the same node mutually regardless of map size.

Table 1. Tree Structure

		Data Set (×1000)					
		1	5	10	20	30	40
Total No. of nodes	R*-tree	119	499	972	2089	3042	3980
	SOM-based R*-tree	51	241	483	928	1300	1705
Heights of nodes	R*-tree	3	4	4	5	5	5
	SOM-based R*-tree	3	4	4	4	4	5
Time cost(sec)	R*-tree	7.55	50.44	111.89	233.89	350.66	476.49
	SOM-based R*-tree	3.27	19.12	40.76	81.95	115.49	153.92

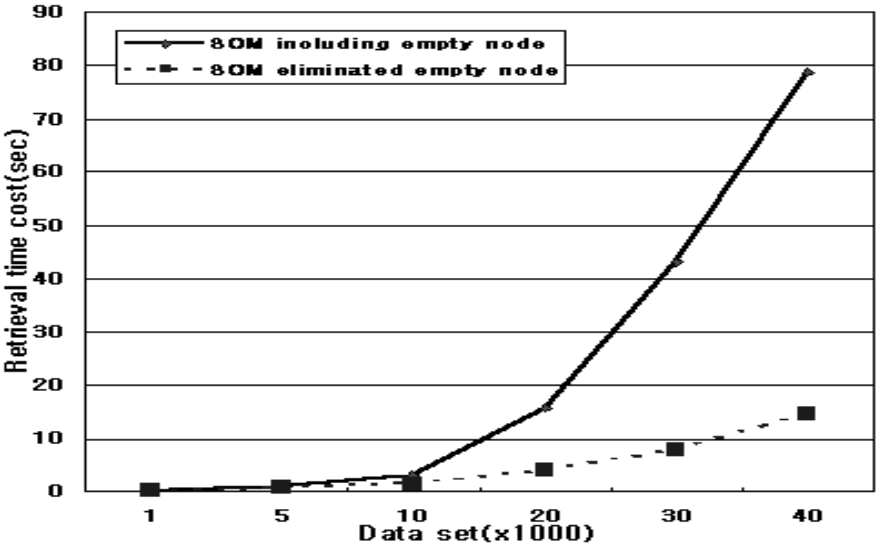


Fig. 3. Retrieval time cost

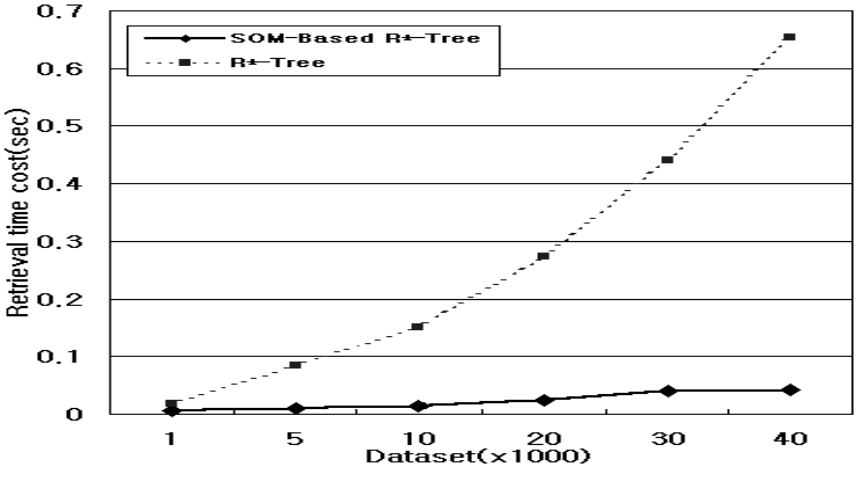


Fig. 4. Comparison of retrieval time cost between SOM-based R*-Tree and R*-Tree

Therefore, reducing the number of nodes and speeding up search time can be realized by eliminating empty nodes; an R*-tree built with this pruned set of nodes will have a smaller overall index size. The height of the tree is not that different, however both the total number nodes and the time cost of building the index decrease. These observations reduce memory usage and retrieval access time. The larger the data set, the more efficient the index.

4.2 Experimental Results

In order to measure search time, we experimented with four types of searches ; search for (i) normal SOM including empty nodes, (ii) normal SOM with eliminated empty nodes, (iii) normal R*-tree, and (iv) SOM-based R*-tree with eliminated empty nodes. The data set size was from 1,000 to 40,000 images. The search method used was the κ-Nearest Neighbor(NN)[15] method, which searches for κ (κ>1) objects nearest to the given query. In SOM, an exhaustive search of the topological feature map is performed, and finding κ (κ=10) nodes nearest to the given query. In the same manner, the normal R*-tree and SOM-based R*-tree are applied using the κ-NN (κ=10) search.

A comparison of retrieval time cost is shown in Figures 3 and 4. In both figures, the horizontal axis is the dataset size. As shown in Figure 3, the retrieval time of SOM with empty nodes, as compared to the SOM without empty nodes, grows drastically as the dataset size increases, over 5 times the retrieval time cost at 40,000 images. Thus, eliminating empty nodes significantly reduces retrieval time by removing unnecessary distance computations. We also compared the performance of the SOM-based R*-tree with that of the R*-tree based on 10-NN retrieval time cost, as shown in Figure 4. In this comparison the nearest OID was obtained for a given query. The retrieval time of the SOM-based R*-tree is far shorter compared to the R*-tree, by 3 to 15 times. The results show that building the R*-tree with overall original feature vectors improves retrieval performance.

Furthermore, the SOM-based R*-tree performs much better than SOM alone, which sequentially searches feature vectors. These experimental results clearly show that a SOM-based R*-tree is more efficient for similarity retrieval.

5 Conclusions

In this study, we proposed SOM-based R*-tree for dealing with similarity retrieval from high-dimensional data sets. Using a topological feature map and a best-matching-image-list (BMIL) obtained via the learning of a SOM, we constructed an R*-tree. The major finding of this study is that building an R*-tree in which the empty nodes in the topological feature map are removed yields an R*-tree with fewer nodes, thus enhancing performance by decreasing unnecessary access, node visits, and overall search times.

In an experiment, we performed a similarity search using real image data and compared the performance of the SOM based R*-tree with a normal SOM and R*-tree, based on retrieval time cost. The R*-tree with fewer nodes experimentally verified to shorter search tome, and search efficiency was improved due to the use of a *k*-NN search, compared to SOM.

References

1. A.F.C.E. Jacobs and D.H Salesin. Fast Multiresolution Image Querying. In Proc. SIGGRAPH95 , pages 6-11, New York, August 1995. ACM SIGGRAPH.

2. C. Faloutsos, W. Equitz, M. Flickner, W. Niblack, D. Petkovic, and R. Barber. Efficient and Effective Query by Image Content. J. of Intell. inform. Syst., 3:231-262,1994.
3. V. N. Gudivada and V. V. Raghavan. Content-based Image Retrieval system. IEEE Computer, 28(9): 18-22, September 1995.
4. Guttman. R-tree: a dynamic index structure for spatial searching. In Proc. of ACM SIGMOD Int. Conf. on Management of Data, pages 45-57,1984.
5. N. Beckmann, H. Kriegel, R. Schneider, and B. Seeger. R*-tree: an efficient and robust access method for points and agement of Data, pages 322-331, Atlantic City,NJ, May 1990.
6. S. Berchtold, C. Bohm, and H.-P. Kriegal. The pyramid technique: towards breaking the curse of dimensionality. In Proc. of ACM SIGMOD int. conf. on Management of data, pages 142-153, Seattle, WA USA, June 1998.
7. K. Chakrabarti and S. Mehrotra. High dimensional feature indexing using hybrid trees. In Proc. of ICDE1999, March 1999.
8. T. Kohonen. Self-Organizing Maps. Springer, Berlin, 1997.
9. T. Kohonen. Self-organizing maps. Proc.of The IEEE, 78(9):1464-1480,1990.
10. M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by Image and Video Content: The QBIC System. IEEE Computer, 28(9):23-32, September 1995.
11. M. Koskelar. Content-Mased Images Retrieval with Self-Organizing Maps. Master's thesis, Helsinki University of Technology, Department of Engineering Physics and Mathematicd, 1999.
12. S.G. Mallat. Multifrequency Channel Decompositions of Images and Wavelet Models. IEEE. Trans., Acoust., Apeech and Signal Proc., 37(12):2091-2110, December 1989.
13. Natsev, R. Rastogi, and K. Shim. WALRUS: A Similarity Retrieval Algorithm for Image Databaese. In Proc. ACM SIGMOD International Conference on Management of Data, pages 396-406, Philadephia, PA, June 1999. ACM SIGMOD.
14. J. C. Russ. The Image Processing Handbook. CRC Press, Boca Raton, 1995.