Pattern Recognition 45 (2012) 4054–4068

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Hypergraph-based image retrieval for graph-based representation

Salim Jouili [a,*], Salvatore Tabbone [b]

[a] EURA NOVA, 4 Rue Emile Francqui, 1435 Mont-St-Guibert, Belgium
[b] Université de Lorraine—LORIA UMR 7503, BP 239, 54506 Vandoeuvre-lès-Nancy Cedex, France

ARTICLE INFO

Article history:
Received 28 November 2011
Received in revised form
17 March 2012
Accepted 16 April 2012
Available online 28 April 2012

Keywords:
Graph indexing
Graph retrieval
CBIR

ABSTRACT

In this paper, we introduce a novel method for graph indexing. We propose a hypergraph-based model for graph data sets by allowing cluster overlapping. More precisely, in this representation one graph can be assigned to more than one cluster. Using the concept of the graph median and a given threshold, the proposed algorithm detects automatically the number of classes in the graph database. We consider clusters as hyperedges in our hypergraph model and we index the graph set by the hyperedge centroids. This model is interesting to traverse the data set and efficient to retrieve graphs.

1. Introduction

Generally, an image retrieval system (IRS) composed of two parts. The first part consists of describing the image: this description may be textual, in this case the image is associated with a set of words (annotations) that describe it. These techniques are widely used in the internet IRS (e.g. google images[1]). In these systems, the set of words that describes the image is potentially extracted automatically from the web page that contains the involved image, such as the file-name of the image, the page title, etc. These words do not always reflect the content of the image, hence the use of a manual annotation of images. Given the skyrocketing number of available images, such an annotation is very expensive. To overcome this problem, some IRS use a visual description of the image. This use pattern recognition techniques to extract important features that will be used as a description of the image in question. This technique is called content-based image retrieval (CBIR). During the last decade, several CBIR systems have been proposed [12,34].

The second part of an IRS is indexing descriptors. The indexing consists of organizing the image descriptors to ensure access as quickly as possible to the relevant images. This part is crucial in any system of information retrieval, particularly the image retrieval. In fact, indexing avoids the sequential search in an image database by direct access to the block (a reduced set) containing the images most similar to the query image. Several indexing methods are used in a robust and efficient way for image retrieval such that methods based on data structures (e.g. B-tree, B++-tree, kd-tree) [3,11,29] and statistical learning [6].

Almost all systems of image retrieval use a statistical representation (feature vector) of the images. The choice of vectors is influenced by their ease of handling (i.e. computation of distances) and the possibility of navigation in the vector space. Indeed, some indexing methods use the principle of partition the Euclidean space to index the vectors representing the images. However, in pattern recognition, the image representation can be broadly divided into statistical and structural methods [7]. In the former, the document is represented by a feature vector, and in the latter, a data structure (e.g. graphs or trees) is used to describe objects and their relationships in the document. The structural representation (e.g. graph) is more powerful than feature vector in terms of representational abilities [22]. The graph structure provides a flexible representation such that there is no fixed dimensionality for objects (unlike vectors), and provides an efficient representation such that an object is modeled by its components and the existing relations between them.

So it is interesting to develop an image retrieval system where images are represented by graphs. Such a system will also have two parts: the first is to extract the graph representing the image, while the second part is to index these graphs. The first part is already well developed in the literature [21]. Nonetheless, only few works have focused on indexing graphs for image retrieval systems. But, this is not surprising because the representability power of graphs cannot fully be exploited due to a lack of computational tools, as it is the case for statistical representation. Recent approaches tend to bridge the gap between the structural and statistical representation by embedding (explicitly [10,15,20,26], implicitly [13,16] or spectrally [24,25,28,38]) graphs into a feature space. However, doing

* Corresponding author. Tel.: +32 491717419.
E-mail addresses: salim.jouili@euranova.eu (S. Jouili), tabbone@loria.fr (S. Tabbone).

[1] http://images.google.fr/.

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2012.04.016

that we loss the structural links and there is no equivalence relation between the two representations. Another way is to use directly the graph representation and to decompose it into more elementary structures and to address them with a multidimensional data representation method. Most of the works using directly graphs are developed in an escalation of the structures of chemical molecules [36]. In more generic works, we distinguish the *GraphGrep* [17] and *gIndex* [41]. Both methods are based on the use of a sub-structure of each graph as its input index (*index features*). The proper functioning of these works requires that the graphs are labeled by discrete values. This is rare for graphs representing images where the labels are generally continuous values that quantify the local characteristics of an image. In addition, these methods are space consuming since we have to store all frequent sub-structures in the involved graph set.

In the present work, we address the problematic of graph indexing using directly the graph domain. We provide a new approach based on the hypergraph model. The main idea of this contribution is first to re-organize the graph space (domain) into a hypergraph structure. In this hypergraph, each vertex is a graph and each hyperedge corresponds to a set of similar graphs. Second, our method uses this hypergraph structure to index the graph set by making use of the centroids of the hyperedges as index entries. By this way, our method does not need to store additional information about the graph set. In fact, our method creates an index that contains only pointers to some selected graphs from the data set which is an interesting feature, especially, in the case of large data sets. Besides indexing, our method addresses also the navigation problem in a database of images represented by graphs. Thanks to the hypergraph structure, the navigation through the data set can be performed by a classical traversal algorithm. The experimental results show that our method provides good performance in term of indexing for tested image databases as well as for a chemical database containing about 35,000 graphs, which points out that the proposed method is scalable and can be applied in different domains to retrieve graphs including clustering, indexing and navigation steps.

2. Hypergraph

A hypergraph is a generalization of a graph, where edges can connect any number of vertices. The hypergraph was defined by Berge [4] and is defined as follow:

Let $H=(\vartheta,\xi)$ be a hypergraph, where $\vartheta=\{x_1,x_2,x_3,\ldots,x_n\}$ is a finite set of vertices and $\xi=\{E_1,E_2,E_3,\ldots,E_m\}$ is a family of subsets of ϑ. We have $E_j \neq \emptyset, \bigcup_{j=1,\ldots,m} E_j=\vartheta$.

ϑ is called the set of vertices, ξ is the set of edges (or hyperedges) and $|\vartheta|$ is the cardinality of H. An edge E_i is represented by a line surrounding its vertices if $|E_i| \geq 2$ (E_1 in Fig. 1), by a loop on the element if $|E_i|=1$ (E_4 in Fig. 1), and by a line joining the two elements if $|E_i|=2$ (E_5 in Fig. 1). If $|E_i|=2$ for all i, the hypergraph becomes an ordinary undirected graph. In a hypergraph, two vertices x_i and x_j are said to be adjacent if there exists an edge E_k, which contains the two vertices ($x_i \in E_k, x_j \in E_k$). Two edges E_i and E_j are said to be adjacent if their intersection is not empty. Every hypergraph has an incidence matrix $(m \times n)$ A_i^j with m columns representing the edges and n rows representing the vertices. The elements in A indicate the membership of vertices to hyperedges as follows:

$$A_i^j = \begin{cases} 1 & \text{if } x_i \in E_j \\ 0 & \text{if } x_i \notin E_j \end{cases}$$

For example, consider the hypergraph $H=(\vartheta,\xi)$ shown in Fig. 1, $\vartheta=\{x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8,x_9,x_{10},x_{11},x_{12},x_{13}\}$ and $\xi=\{E_1,E_2,E_3,E_4,E_5,E_6\}$. The cardinality of this hypergraph is $|\vartheta|=13$, the incidence

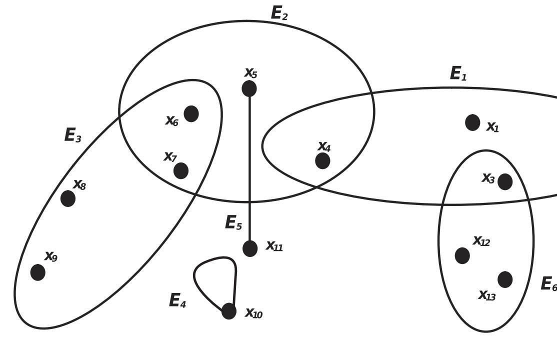

Fig. 1. Example of a hypergraph.

matrix is defined as:

	E_1	E_2	E_3	E_4	E_5	E_6
x_1	1	0	0	0	0	0
x_2	1	0	0	0	0	0
x_3	1	0	0	0	0	1
x_4	1	1	0	0	0	0
x_5	0	1	0	0	1	0
x_6	0	1	1	0	0	0
x_7	0	1	1	0	0	0
x_8	0	0	1	0	0	0
x_9	0	0	1	0	0	0
x_{10}	0	0	0	1	0	0
x_{11}	0	0	0	0	1	0
x_{12}	0	0	0	0	0	1
x_{13}	0	0	0	0	0	1

Recently, the hypergraph has been used in the pattern recognition domain, for object representation [25], similarity measures [8], and object clustering [1,5,37].

3. Hypergraph-based model

It is important to organize the graphs in coherent sets to facilitate later indexing. Such organization can be done with a unsupervised classification technique. However, the use of classification for indexing requires changes in the strategy of classifiers. Indeed, in traditional approaches of (un-)supervised classification, an object o is always assigned to one and only one class c for which the object o is more similar to the objects in the class c than the objects in other classes. Obviously, this similarity is based on all the characteristics of each object. In general, to classify an object into one class among k classes; first the k distances between the object and the k classes are calculated, then the object is assigned to the class with the minimum distance. This strategy is retained even if the differences between these distances is very low (see Fig. 2). This assignment is determined by the fact that the distance d_2 between o and C_2 is less than the distance d_1 between o and C_1. In this illustration, we see that the two distances d_1 and d_1 are very similar, and the object has been assigned to C_2. In the case where the objects are graphs, we consider that this strategy can constrain the indexing. Given a set of objects, the indexing is based on the set of classes $C=\{c1,\ldots,c_n\}$ arising from a classical classification, the search for similar objects to a query object o_r provides direct access to the nearest class C_l to o_r. Thus, retrieval of all objects similar to a query o_r is limited to the objects belonging to C_l, i.e. all other classes are omitted. On the contrary, it is likely that objects do not belong to C_l are similar to o_r.