Pattern Recognition ▮ (▮▮▮▮) ▮▮▮–▮▮▮

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

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 *Graph-Grep* [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 matrix is defined as:

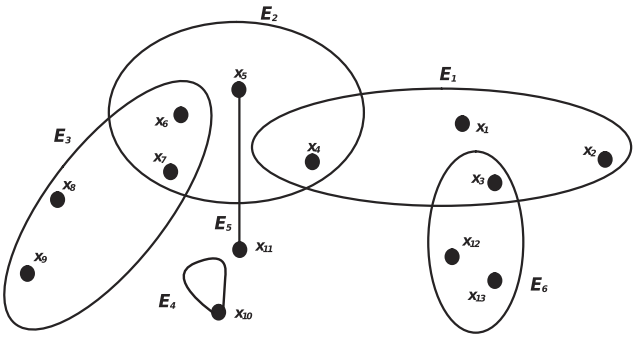

Fig. 1. Example of a hypergraph.

	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.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

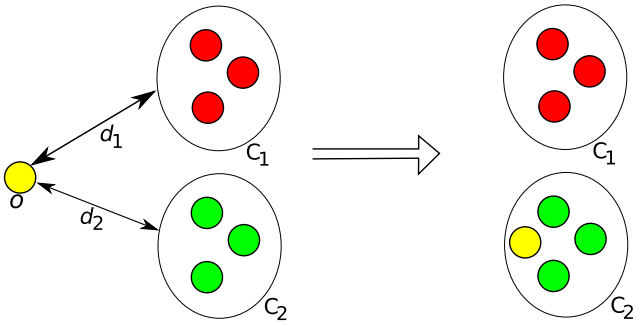

Fig. 2. Illustration of an object classification.

In the remainder of this paper, we propose a solution to this problem based on the structure of the hypergraph. The idea is to represent a set of graphs into a hypergraph structure where the vertices correspond to graphs and the hyperedges correspond to coherent graph sets (clusters).

Thus, in our work we assume that one graph can belongs to several clusters, so we consider that the proposed hypergraph is connected (*1-edge-connected*). Therefore, each graph G_i in the proposed structure is assigned to $\Delta_\vartheta(G_i)$ clusters and each cluster C_j contains $\Delta_\xi(C_j)$ graphs. However, a key problem in structuring a set of graphs into a hypergraph is the determination of the number of clusters (hyperedges) and the determination of related graphs (similar graphs) that can be grouped as hyperedges. In this perspective, we consider that the number of hyperedges is equal to the size of a representative set, defined on the selection of the most representative graphs into the whole set. We denote each selected graph as a hyperedge centroid. The selection of these graphs is similar to the problem of Prototype Selection [2,26,35]. Riesen et al. [26] enumerate some techniques to select prototypes from a training set. These techniques require a specification of the number of prototypes and there are no premises for determining automatically this number. Therefore, if we are in a unsupervised context where no information about the number of representative graphs is available, this number will be determined empirically. In this perspective, Spath [35] proposes an algorithm using leaders and distance-based threshold where the number of selected prototype is inversely proportional to the selected threshold. However, the Leader algorithm [35] is sensitive to the selection of the initial prototype which is selected randomly among the input data. To overcome this problem, we introduce a representative graphs (hyperedge centroids) selection based on a peeling-off strategy. This method can be viewed as an improvement of the Leader and the *K*-Centers algorithms. After the selection of the hyperedge centroids, we define the hypergraph structure by assigning each graph to the corresponding hyperedges. Then the browsing and the retrieval of the graphs will be transposed into the hypergraph structure.

4. Hypergraph-based indexing

In this section, we present the main three parts of our hypergraph model. Each part of the proposed method can be considered as an independent contribution. In fact, we propose, first, an extension and an improvement of a well-known prototype selection algorithm for the graph domain. Second, we introduce a procedure to re-organize a set of graphs into a hypergraph structure by means of the prototype selected in the first contribution. Finally, we present a graph retrieval algorithm that use the new hypergraph structure. In addition to classical retrieval, we show that our method can be used for an associative retrieval purpose in which a given user can explore and navigate within the graph data sets. Fig. 3 illustrates the

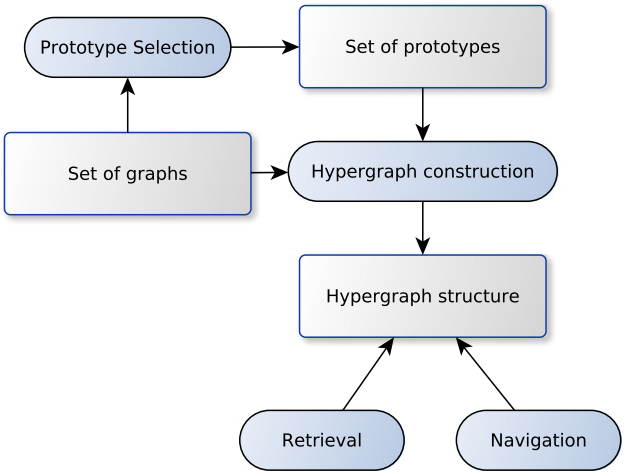

Fig. 3. Illustration of the workflow of the proposed method.

working model of our method which can be described as follow: starting from a set of graphs, the new prototype graph selection is applied to select the most representative graphs of the whole set. Then, the initial graph sets is organized into a hypergraph structure by means of the prototypes. Finally, the hypergraph structure is used for retrieval and exploring purposes.

4.1. Hyperedge centroids selection

As stated above, the hyperedge centroids selection is similar to the Prototype Selection problem. Therefore, we aim to select a set of graphs which capture the most significant aspects of a set of graphs. In this purpose, we introduce an improvement for the Leader algorithm [35]. The proposed algorithm proceeds as follows:

1. Select the median graph [40] G_m from the unassigned graphs in the whole set of graphs S. Then the furthest graph G_{p_k} (which has not been previously assigned) to G_m, becomes the centroids of the cluster C_k. In the first iteration, the graph G_{p_k} is the initial selected prototype.
2. Distances of every unassigned graph $g_i \in S \backslash \{G_{p_k}\}$ are compared with that of the last selected prototype G_{p_k}. If the distances $d(g_i, G_{p_k})$ and $d(g_i, g_{j \in C_k})$ are less than a user-specified threshold T, the graph g_i is assigned to the cluster C_k with the centroid G_{p_k}, and g_i is tagged as assigned.
3. Recompute the median graph G_{m_k} of C_k, if $G_{m_k} \neq G_{p_k}$, replace G_{p_k} by G_{m_k}. If any replacements is done, go to the next step, otherwise all g_j are tagged as unassigned, $\forall g_j \in C_k$, then return to step 2.
4. While S contains an unassigned graphs return to step 1, otherwise stop.

A first improvement consists in adapting the algorithm of the leader in the space of graphs using the notion of median graph. Then a new method of selecting the initial prototype has been developed. In the algorithm of the Leader, the choice of the initial prototype is done randomly, which affects the final result of clustering (i.e. at each execution results vary). We chose the farthest graph from the median of the whole set as the initial prototype of the algorithm. This choice guaranteed, therefore, the stability of results. A second improvement consists of providing a new iterative selection of prototypes. In fact, as shown in the pseudo-code of the algorithm, once the initial prototype is selected we consider all graphs which have a distance to the prototype less than or equal to a given threshold, in addition, the distance between them

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

(the considered graphs) must be also less than or equal to the same threshold. In contrast, the leader algorithm considers only the distances between the prototype and each object (there is no criterion on distances between objects).

Given a threshold T, the algorithm clusters the set of graphs with an intra-class inertia (I_i) less or equal to T. This property is performed on the step 2. In addition, this algorithm ensures the selection of the prototypes which are given by the centers of the resulted clusters. Furthermore, it guarantees a certain separability between classes of one partition. By using an edit distance d, we can formulate the between-class inertia (I_b) of a partition C composed of two classes C_1, C_2 by the Ward [39] criterion:

$$I_b(C_1,C_2) = \frac{\eta_1 \times \eta_2}{\eta_1+\eta_2} d^2_{g_{c1},g_{c2}} \quad (1)$$

where g_{c_i} is the centroid of the class C_i and η_i is the number of members of C_i. The analysis of this formula shows that there is a strong dependence between the interclass inertia and the centroid. Indeed, we know in our case that the distance between two centroids is higher than the threshold T and $I_i \leq T$.

4.2. The hypergraph-based representation

Let S be the whole set of graphs and P be the set of selected prototypes P ($P \subset S$). Classical clustering techniques find for each graph $g \in S \backslash P$ its nearest neighbor $p_i \in P$ and add the graph to the cluster C_i corresponding to the prototype p_i. In fact, if a graph g presents a similar distances to two prototypes p_i and p_j, g is added to the cluster with the nearest prototype even though the difference between the two distances is very minor. Moreover, the provided clusters are disjoint and can be exploited for a retrieval task as used in [27,30–32], but it will be difficult to find an algorithm for browsing the whole set of graphs through disjoint clusters.

On the contrary, a hypergraph-based model allows the overlapping of clusters. Henceforth the clusters will be viewed as hyperedges of hypergraph and the graphs as the vertices. First, for each selected prototype p_i a hyperedge h_i is defined with a centroid p_i. Second, every hyperedge is defined as follows : each graph $g \in S \backslash P$ is added to the hyperedges with the nearest prototypes to g (their distances to g is less than the threshold T used in the previous algorithm). From this definition, we can conclude the following theorem:

Theorem. *Let $d(.,.)$ be a metric graph distance and let $\mathcal{H}$ be the hypergraph generated with a given threshold T. If a graph g is shared by two hyperedges $h_i \in \mathcal{H}$ and $h_j \in \mathcal{H}$ with the centroids p_i and p_j, respectively, then $T < d(p_i,p_j) < 2 \times T$.*

Proof. Let P be the set of selected prototypes, so

$$\forall p_i,p_j \in P, \quad d(p_i,p_j) > T \quad (2)$$

Since the distance between graphs is metric, then from the triangular inequality:

$$d(p_i,p_j) \leq d(p_i,g) + d(g,p_j) \quad (3)$$

We know that $g \in h_i$ and $g \in h_j$, so

$$d(p_i,g) < T \quad (4)$$

and

$$d(g,p_j) < T \quad (5)$$

As the threshold T and the distances are positive, from (3) and (4) we can write:

$$d(p_i,g) + d(g,p_j) < 2 \times T$$

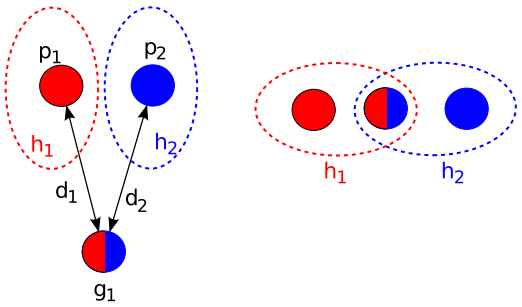

Fig. 4. Illustration of the proposed model. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

which indicates (from (3)) that,

$$d(p_i,p_j) < 2 \times T \quad (6)$$

Finally, by combining (2) and (6), we find

$$T < d(p_i,p_j) < 2 \times T. \quad \square$$

This theorem shows that if the used graph distance is a metric, the overlap is achieved only between neighbouring hyperedges. Specifically, the hyperedges, that share common graphs, have nearby centroids. This property provides an interesting semantic aspect of the method which can be summarized in the following formula: "*When a graph sharing information with two (or more) close clusters, it must be assigned to all these clusters*". Obviously, this theorem is not always valid if the graph distance is not metric.

Fig. 4 illustrates our motivation. In the leftmost part of the figure, we suppose that d_1 and d_2 are less or equal than T, so the graph g_1 shares some information with p_1 and p_2 (information are illustrated in colors). With the hypergraph model, we will be able to assign g_1 to the both hyperedges h_1 and h_2. The rightmost part of Fig. 4 describes how two hyperedges (clusters) can overlap with one graph in common. Here, $\Delta_\vartheta(g_1) = 2$ and $\Delta_\xi(h_1) = \Delta_\xi(h_2) = 2$.

Once all hyperedges are defined from the graph set, we recompute, for each hyperedge, the median graph which will be the new centroid of the hyperedge. The aim of this step is to update the hyperedge centroids and to maintain as much information as possible about the graphs in the corresponding hyperedge. We choose the median graph to define the centroid of a cluster because, unlike the generalized median graph and the minimum common super-graph [9], it needs less computation time. We also use this technique when adding new graphs in the database. Specifically, each new graph g_n is added to the hyperedges with centroids near to g_n (their distance to g_n is less than the threshold T used above). Then, we change the centroid of each affected hyperedge by the new median graph. The hyperedges affected by the update of centroid are those where the graph g_n is added.

4.3. Image retrieval

Smeulders et al. [34] define three paradigms in the image retrieval according to the user's objectives:

- Associative retrieval: Users of associative retrieval have not really defined goal. The objective of this image retrieval paradigm is to help users to explore the image database. Associative retrieval often involves the refinement of all the images presented initially to the user, via interactions.
- Target retrieval: This paradigm regards the users who aim to retrieve a specific image, such as finding a picture of a specific object in a museum collection.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

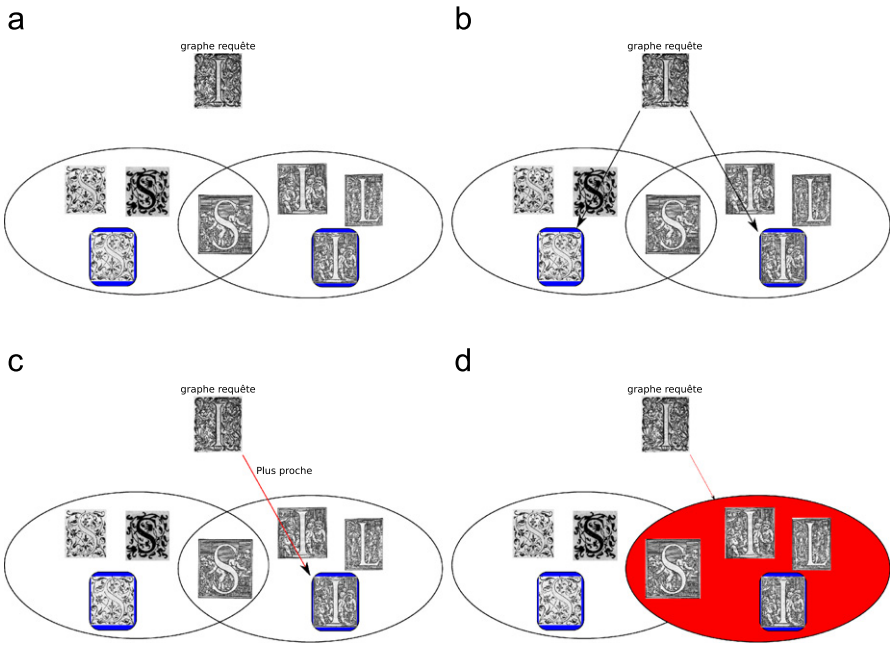

Fig. 5. Illustration of our retrieval method. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

- Category retrieval: This paradigm consists of retrieving the maximum amount of images that belonging to a specific class defined by the user. It may correspond to the case where the user has an example (image) and looking for other images of the same class. Categories can be derived from tags or from the database using adequate similarity measures.

Regarding graphs, querying a set of graphs consists usually of finding the most similar graphs to a given query. This retrieval process sorts the distances between the query and the graphs of the database in an ascending order. This type of retrieval corresponds to the second paradigm of Smeulders et al. [34] when the user aims to find a specific image (represented by a graph). In this case, the desired image is the image represented by the closest graph to the graph query. Moreover, the classic interrogation of a set of graphs corresponds also to the third paradigm of Smeulders et al. [34]. The user aims to look for the most similar images (represented by graphs) to its query. Nevertheless, the ubiquity of noise in the images do not promote the use of classical querying of graphs for category retrieval. Indeed, the noise causes changes in the corresponding graphs which often leads to false positive results. In the literature, some authors [27,30–32] consider this problem as a consequence of the lack of efficient exploitation of the distribution of distances between graphs. Indeed, to improve the retrieval performance of *category*, some works (e.g. [27]) propose some retrieval techniques based on clustering of graphs to fully exploit the distribution of distances. In these works, similar images to the query are retrieved in the cluster closest to the query image rather than searching into the entire database. This requires a preliminary phase of clustering of the image database.

In the same vein, we introduce a procedure that involves the proposed hypergraph model. The main idea is to find the nearest centroid (among all hyperedge centroids) to a given query. Then, we look for the most similar graphs within the hyperedge which it centroid is the most similar to the query. We can describe the retrieval procedure into the hypergraph model as follows:

1. For a query graph g_q, compute the set of distances between g_q and each hyperedge centroid.
2. Initialize to empty the set $\mathcal{R}=\emptyset$ of the retrieved graphs.
3. Get the nearest hyperedge centroid p_i to g_q. Let h_i be the hyperedge with the centroid p_i.
4. Retrieve the set $\mathcal{R}=\mathcal{R}\cup\{g_j\}$ containing the most similar graphs g_j to g_q, where $g_j \in h_i$.
5. If the number of retrieved graphs is not sufficient or the desired graph is not found, go to the next step. Otherwise, end the algorithm.
6. Get p_j the next closest centroid to g_q and h_j the hyperedge with the centroid p_j. Return to Step 3 by considering the next hyperedge (h_j), i.e. $h_i \leftarrow h_j$.

Fig. 5 shows an illustration of this retrieval method. Indeed, for a given query (Fig. 5(a)), we start by identifying the nearest centroid (Fig. 5(b) and (c)). Then, the retrieval is achieved in the hyperedge containing this centroid (Fig. 5(d)). In this example, we consider that the response to the query is satisfied by the images of the first hyperedge (red). Moreover, if this was not the case, the algorithm will retrieve also the images in the second hyperedge.

Our contribution can also be used in an associative retrieval. We recall that this paradigm of image retrieval consists of helping users to explore an image database. To navigate into the image database, we use a technique of modified depth-first traversal of the hypergraph. Our method consists on using the centroids of hyperedges as the start points of navigation. The user then chooses the hyperedge to explore by selecting the corresponding centroid. Then, the navigation is a depth-first traversal of the hypergraph. This traversal corresponds to a visit of the images belonging to connected hyperedges. It should be noted here that the traversal of the hypergraph depends on the user's choice. Indeed, at one stage e_i of navigation, the traversal is defined from an image *im* chosen by the user. Thus, in the next step e_{i+1}, the explored part of the hypergraph corresponds the nearest images to *im*.[2] Specifically, the navigation of the hypergraph is achieved through a graphical user interface (GUI). In this interface, the user

[2] The nearest images to an image are into the hyperedges containing the image *i*.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

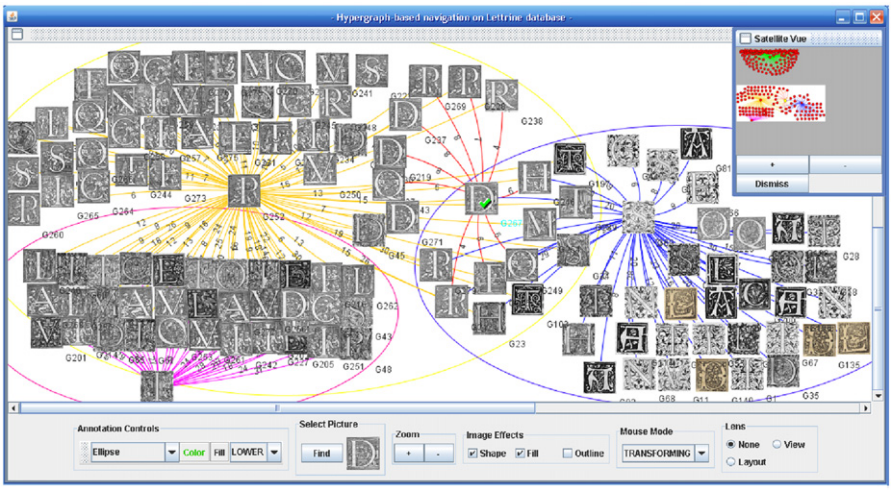

Fig. 6. A navigation GUI.

makes his traversal from the hypergraph by clicking on the images that interest him. Fig. 6 illustrates a navigation interface where the hyperedges are represented by ellipses that overlap. In this figure, the user selects the drop cap with the letter D, by clicking on the adjacent initials he is able to explore a part of the database related to its query. The images in the center of each hyperedge are the centroids.

5. Experimental results

5.1. Graph similarity measure by means of node signatures

Before going ahead on the experimental parts, let us recall how the similarity can be computed in the domain of graphs. Similarity or (dissimilarity) between two graphs is almost always referred as a graph matching problem. Graph matching is the process of finding a correspondence between nodes and edges of two graphs that satisfy some constraints ensuring that similar substructures in one graph are mapped to similar substructures in the other. Many approaches have been proposed to solve the graph matching problem. In this paper we use a recent technique proposed by Jouili et al. [19] but any other graph matching methods can be used. This approach is based on node signatures notion. In order to construct a signature for a node in an attributed graph, all available information into the graph and related to this node is used. The collection of these information should be refined into an adequate structure which can provides distances between different node signatures. In this perspective, the node signature is defined as a set composed by four subsets which represent the node attribute, the node degree and the attributes of its adjacent edges and the degrees of the nodes on which these edges are connected. Given a graph $G=(V,E,\alpha,\beta)$, the node signature of $n_i \in V$ is defined as follows:

$$\gamma(n_i)=\{\alpha_i,\theta(n_i),\{\theta(n_j)\}_{\forall ij \in E},\{\beta_{ij}\}_{\forall ij \in E}\}$$

where

- α_i the attribute of the node n_i.
- $\theta(n_i)$ the degree of n_i.
- $\{\theta(n_j)\}_{\forall ij \in E}$ the degrees set of the nodes adjacent to n_i.
- $\{\beta_{ij}\}_{\forall ij \in E}$ the attributes set of the incident edges to n_i.

Then, to compute a distance between node signatures, the *Heterogeneous Euclidean Overlap Metric* (HEOM) is used. The HEOM uses the *overlap* metric for symbolic attributes and the normalized Euclidean distance for numeric attributes. Next the similarities between the graphs is computed as follows: First, a definition of the distance between two sets of node signatures is given. Subsequently, a matching distance between two graphs is defined based on the node signatures sets. Let S_γ be a collection of local descriptions, the set of node signatures S_γ of a graph $g=(V,E,\alpha,\beta)$ is defined as:

$$S_\gamma(g)=\{\gamma(n_i)\,|\,\forall n_i \in V\}$$

Let $A=(V_a,E_a)$ and $B=(V_b,E_b)$ be two graphs. And assume that $\phi : S_\gamma(A) \rightarrow S_\gamma(B)$ is a function. The distance d between A and B is given by φ which is the distance between $S_\gamma(A)$ and $S_\gamma(B)$

$$d(A,B)=\varphi(S_\gamma(A),S_\gamma(B))=\min_{\phi} \sum_{\gamma(n_i) \in S_\gamma(A)} d_{nd}(\gamma(n_i),\phi(\gamma(n_i)))$$

The calculation of the function $\varphi(S_\gamma(A),S_\gamma(B))$ is equivalent to solve an assignment problem, which is one of the fundamental combinatorial optimization problems. It consists of finding a maximum weight matching in a weighted bipartite graph. This assignment problem can be solved by the Hungarian method [23]. The permutation matrix P, obtained by applying the Hungarian method to the cost matrix, defines the optimum matching between two given graphs.

5.2. Setup

In this experimental part, we focus on studying the behavior of our approach with respect to the used threshold T. This study covers, for a given value of T, the following five points:

1. The number of hyperedges generated by our method: This aspect describes the relationship between the number of clusters detected in a database of graphs and the threshold value.
2. The average size of hyperedges: This aspect describes the evolution of the average size of clusters, provided by our method and compared to the threshold value.
3. The overlapping rate: The overlapping between two hyperedges h_i and h_j corresponds to the set $\mathcal{G}$ containing the graphs shared by h_i and h_j, i.e. $\mathcal{G}=\{g_i\,|\,g_i \in h_i \text{ and } g_i \in h_j\}$. To compute the overlapping rate of our hypergraph structure, we use

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

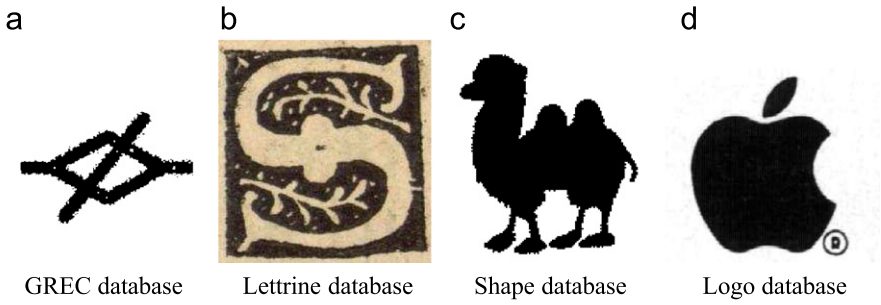

Fig. 7. Datasets.

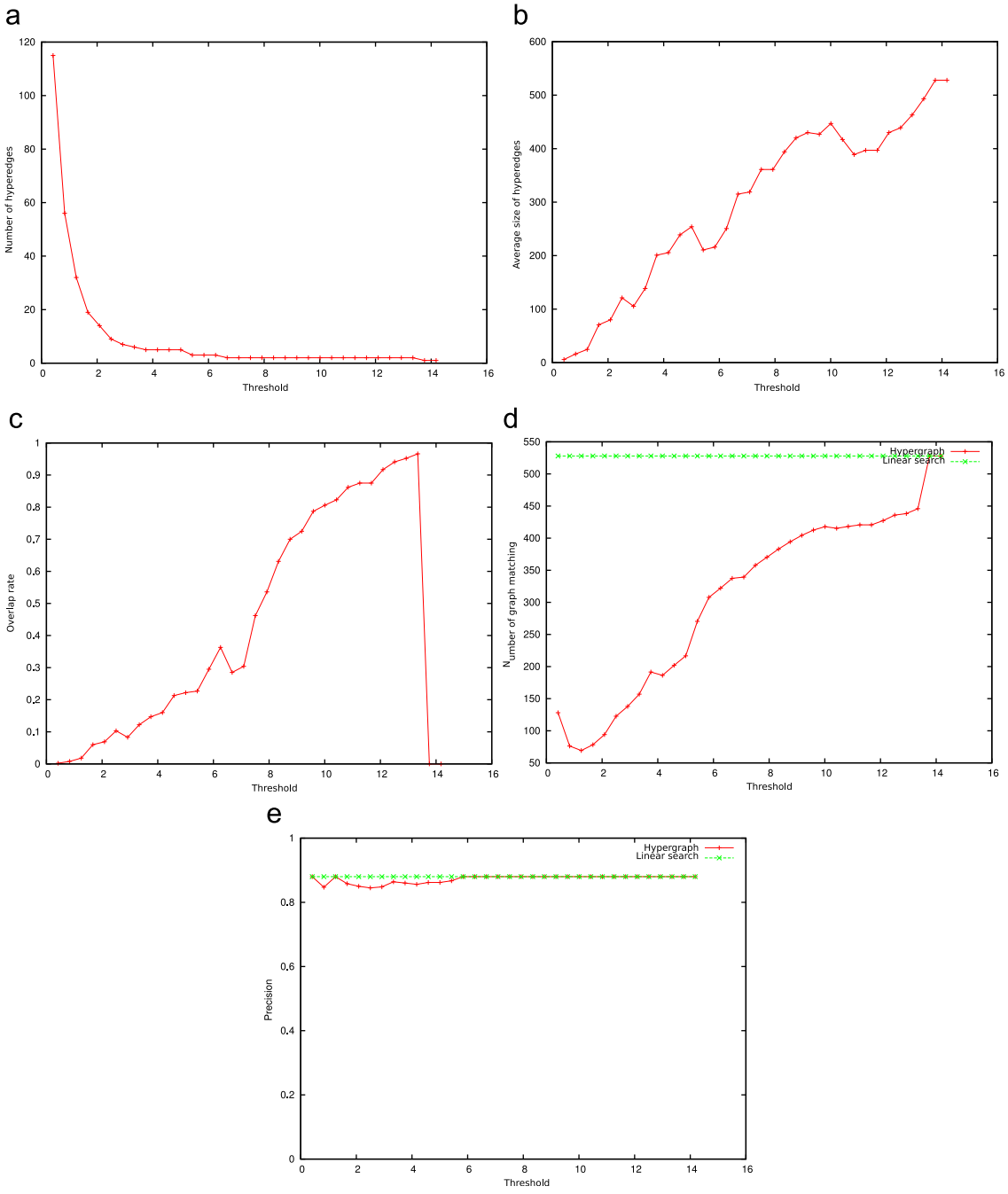

Fig. 8. Behavior of our hypergraph structure according to the threshold value: database GREC1: (a) number of hyperedges generated by our method, (b) the average size of hyperedges, (c) the overlapping rate, (d) number of graph matching and (e) retrieval precision.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

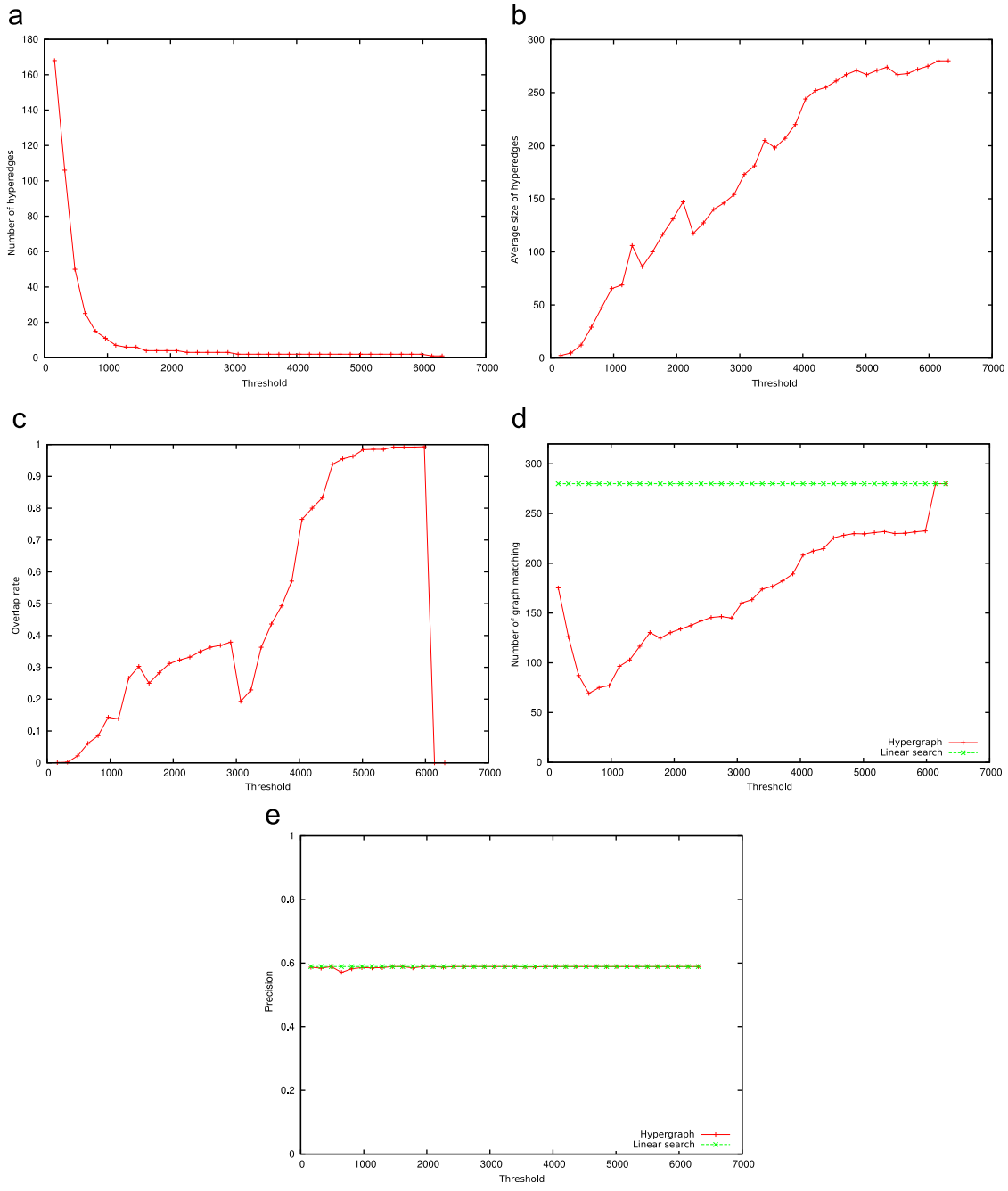

Fig. 9. Behavior of our hypergraph structure according to the threshold value: database Lettrine: (a) number of hyperedges generated by our method, (b) the average size of hyperedges, (c) the overlapping rate, (d) number of graph matching and (e) retrieval precision.

the measure of Hong et al. [18]. Note that this measure was developed in the context of fuzzy clustering which allows overlapping between clusters. Specifically, the rate of overlapping between two hyperedges h_i and h_j ($\tau(h_i,h_j)$) is the quotient of the number of shared graphs divided by the number of graphs of the smallest hyperedge. Formally,

$$\tau(h_i,h_j) = \frac{N_{shared}}{N_{min}}$$

with N_{shared} the number of shared graphs between h_i and h_j, and $N_{\min} = \min(|h_i|,|h_j|)$ is the size (number of graphs) of the smallest hyperedge between h_i and h_j. The rate of overlapping is in the interval [0, 1] (value close to 1 indicates a strong overlapping).

The final overlapping rate of a hypergraph $\mathcal{H}$ ($\tau(\mathcal{H})$) is the average of all overlapping rates between pairs of hyperedges in the hypergraph, or formally:

$$\tau(\mathcal{H}) = \frac{\sum \tau(h_i,h_j)}{\frac{n(n-1)}{2}}$$

with n is the number of hyperedges in $\mathcal{H}$.

4. The number of graph matching: This aspect determines the number of graph matching performed on an image retrieval procedure.
5. The retrieval precision: This aspect assesses the quality of image retrieval performed by our method. Precision is the quotient of the number of relevant images found by the total

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

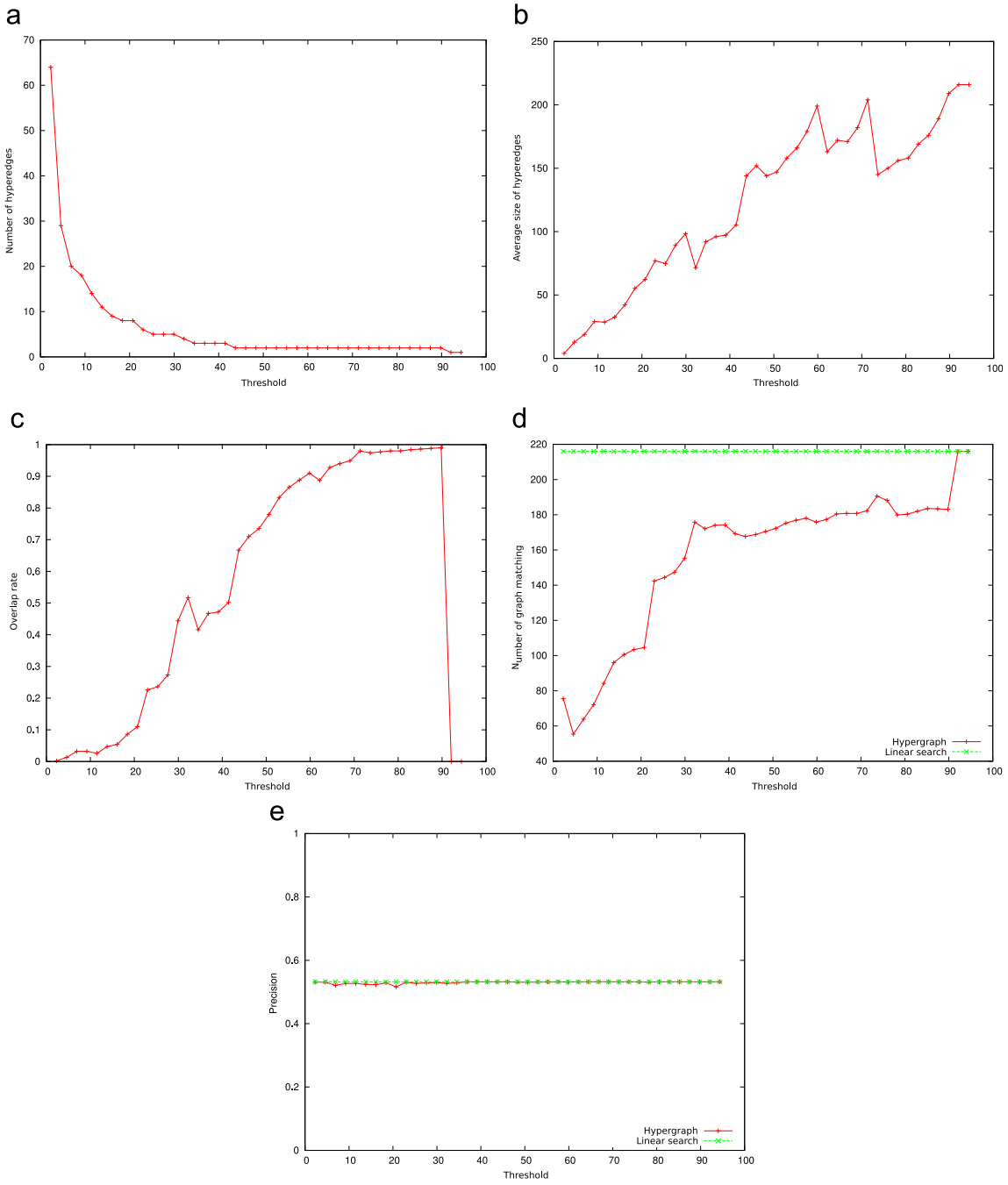

Fig. 10. Behavior of our hypergraph structure according to the threshold value: database Shape: (a) number of hyperedges generated by our method, (b) the average size of hyperedges, (c) the overlapping rate, (d) number of graph matching and (e) retrieval precision.

number of retrieved images by our proposed method for a given query.

To evaluate our method, we used the following datasets:

- *GREC1 database*: The GREC1 database[3] (see Fig. 7(a)) consists of graphs representing symbols from architectural and electronic drawings. Here, the ending points (i.e. corners, intersections and circles) are represented by nodes which are connected by undirected edges. The graph database used in our experiments has 528 graphs, 22 classes and 24 graphs per class.
- *Lettrine database*: This database (see Fig. 7(b)) contains lettrine (graphical object) extracted from digitized ancient documents.[4] Since one lettrine contains a lot of information (i.e. texture, decorated background, letters), the graphs are extracted from a region-based segmentation [14] of the lettrine with a user-based parametrization technique. The nodes of the graph are represented by the regions and the edges describe their adjacency relationships. The graph database used in our experiments consists of 280 graphs, 4 classes and 70 graphs per class.

[3] http://www.cvc.uab.es/grec2003/SymRecContest/index.htm.

[4] Provided by the CESR—University of Tours on the context of the ANR Navidomass project http://l3iexp.univ-lr.fr/navidomass/.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

- *Shape database*: We use the shapes provided by the LEMS laboratory of the Brown University [33] (see Fig. 7(c)). The graphs are extracted from the shapes by skeletonizing and applying a polygonal approximation on the skeleton to obtain straight line segments. For each line segment, we locate endpoints and the graphs are based on the Delaunay triangulations of these endpoints. The graph database used in our experiments has 216 graphs, 18 classes and 12 graphs per class.
- *Logo database*: This database (see Fig. 7(d)) consists of graphs representing binary images of trademark-logos. Here, the region adjacency graphs are used to represent the logos. The graph database used in our experiments consists of 80 graphs, with 10 classes and eight graphs per class.
- *Mutagenicity*: This database consists of 400 graphs (two classes) representing molecular compounds. The nodes represent the atoms labeled with the corresponding chemical symbols and edges by valence of linkage.
- *Letter*: This database (750 graphs, 15 classes) involves graphs that represent distorted letter drawings. Each distorted letter corresponds to a graph by representing lines by edges and ending points of lines by nodes. The nodes are labeled by two-dimensional attributes giving its position.
- *GREC2*: This database (528 images, 22 classes) consists of graphs representing symbols from architectural and electronic drawings. Here ending points (ie corners, intersections and circles) are represented by nodes which are connected by undirected edges and labeled as lines or arcs.

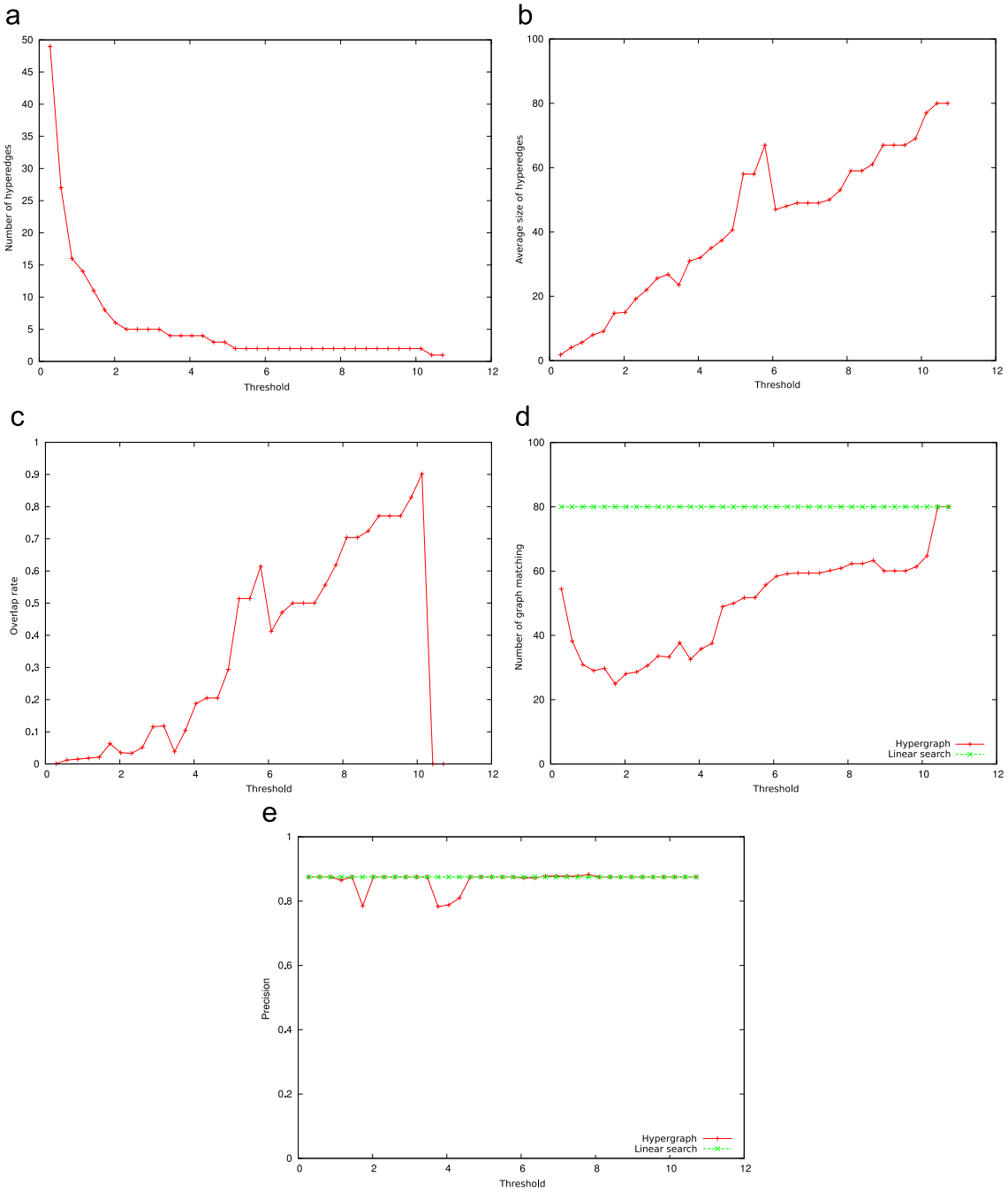

Fig. 11. Behavior of our hypergraph structure according to the threshold value: database Logo: (a) number of hyperedges generated by our method, (b) the average size of hyperedges, (c) the overlapping rate, (d) number of graph matching and (e) retrieval precision.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

For each graph database, we compute the five criteria, defined above, for several values of the threshold T. Regarding the retrieval precision and the number of graph matching, we compute, for each database, the average precision (respectively, the number of matching average) on the first ten images selected by our method and where each image of the database is set as a query image. We recall that in the case where the number of graphs in the first hyperedge is less than ten, our method retrieves the graphs of the next nearest hyperedge.

We compare the precision and the number of graph matching made by our method with those of the classical method of image retrieval in which the query is compared with all graphs in the database. We provide a comparison with only the exhaustive retrieval because, to the best of our knowledge, the previous graph indexing works do not deal with graphs with non-discrete label values and do not provide a way to navigate within a graph data sets.

5.3. Results

Figs. 8–14 show, respectively, the behavior of our method for the different databases. Each figure is composed of five curves where each one shows the behavior of one performance of our method related on the threshold value. From these results we can draw the following remarks:

- Considering the criteria of the number and the average size of hyperedges generated based on the threshold, we observe that the curves have the same appearance (their slopes are roughly the same) for all databases. This appearance shows that for

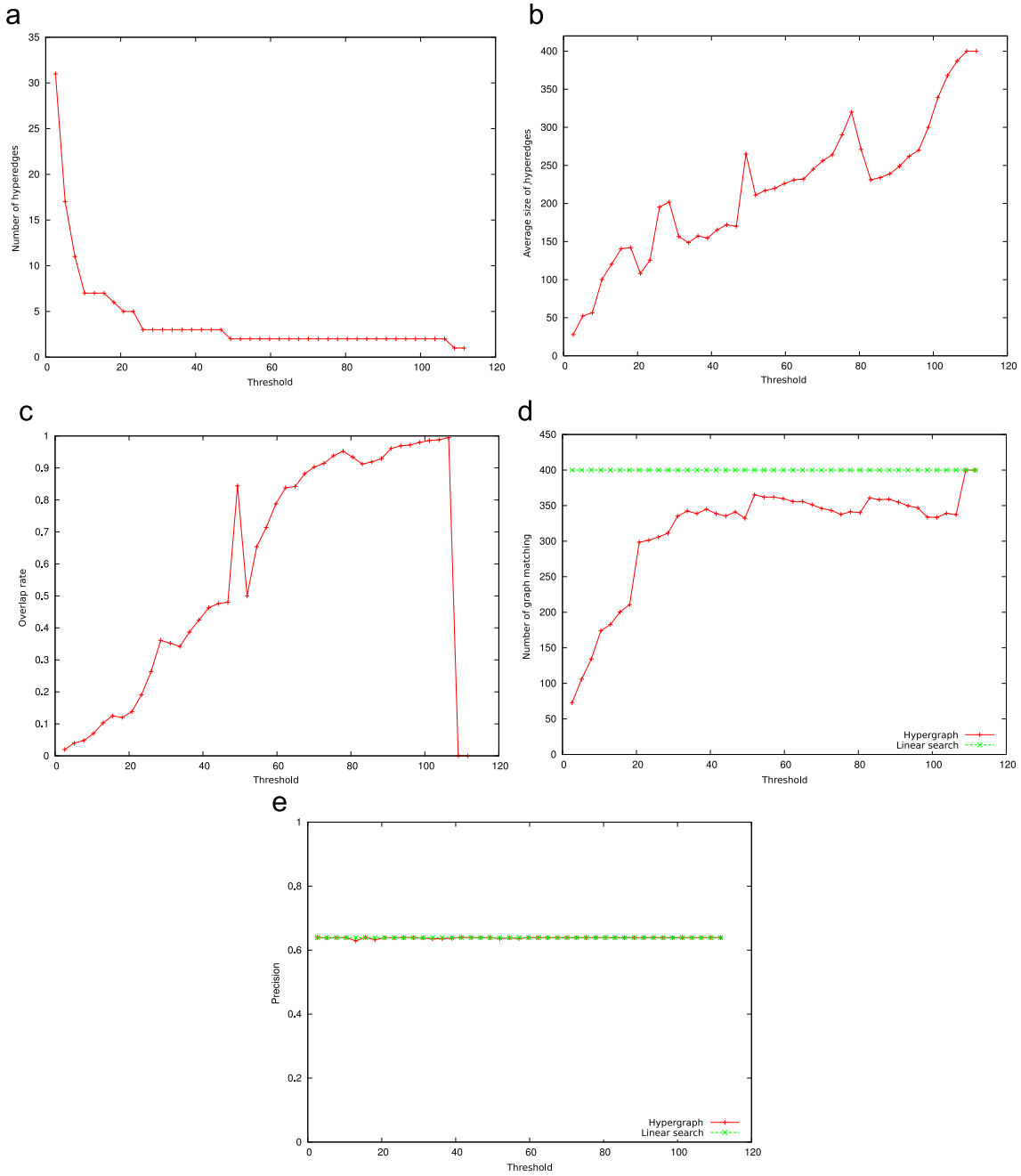

Fig. 12. Behavior of our hypergraph structure according to the threshold value, database mutagenicity: (a) number of hyperedges generated by our method, (b) the average size of hyperedges, (c) the overlapping rate, (d) number of graph matching and (e) retrieval precision.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

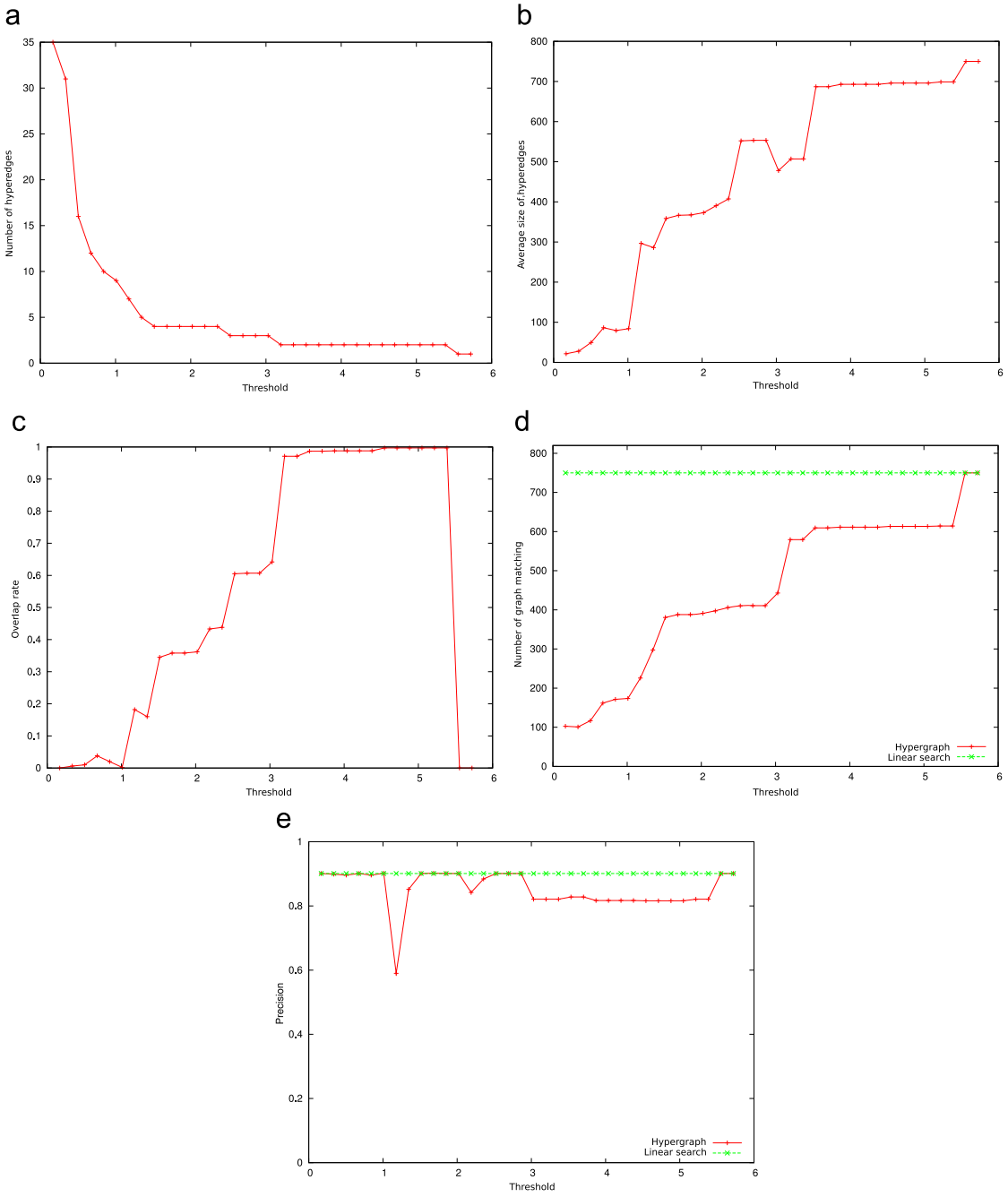

Fig. 13. Behavior of our hypergraph structure according to the threshold value, database Letter: (a) number of hyperedges generated by our method, (b) the average size of hyperedges, (c) the overlapping rate, (d) number of graph matching and (e) retrieval precision.

small values of the threshold, the number of hyperedges is important and their average size is small. Indeed, for a small threshold value, each hyperedge contains a small set of graphs very similar (their distance is below the given threshold). On the contrary, for large values of the threshold, the method generates few hyperedges but with large sizes.

- Considering the overlapping rate, we observe also that the curves show the same appearance for all databases. Indeed, higher values of the threshold imply a higher overlapping rate. In fact, for large values of the threshold, graphs are more likely to belong simultaneously to several hyperedges. Moreover, if the threshold exceeds the distance between the two most distant graphs in a database, our method generates a single hyperedge that contains the entire database. This is explained by the fact that all distances between graphs are less than the threshold. In this case, the overlapping rate fall down to zero.
- Considering the number of graph matching, we observe that when the threshold is high the number of matching is similar to the linear search. In this case the structure tends to one hyperedge and the query is matched with all the graphs of the database as it is the case for a linear search. Also, we can remark when the threshold is too small and tends to zero the number of matching is near to the linear search. In this case the structure contains much as hyperedges as graphs. In this case a query is matched with each hyperedge which is similar to match the query with each graph. Furthermore, related to the retrieval precision, we note that our method achieves the same degree of precision than classical retrieval (linear search).

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

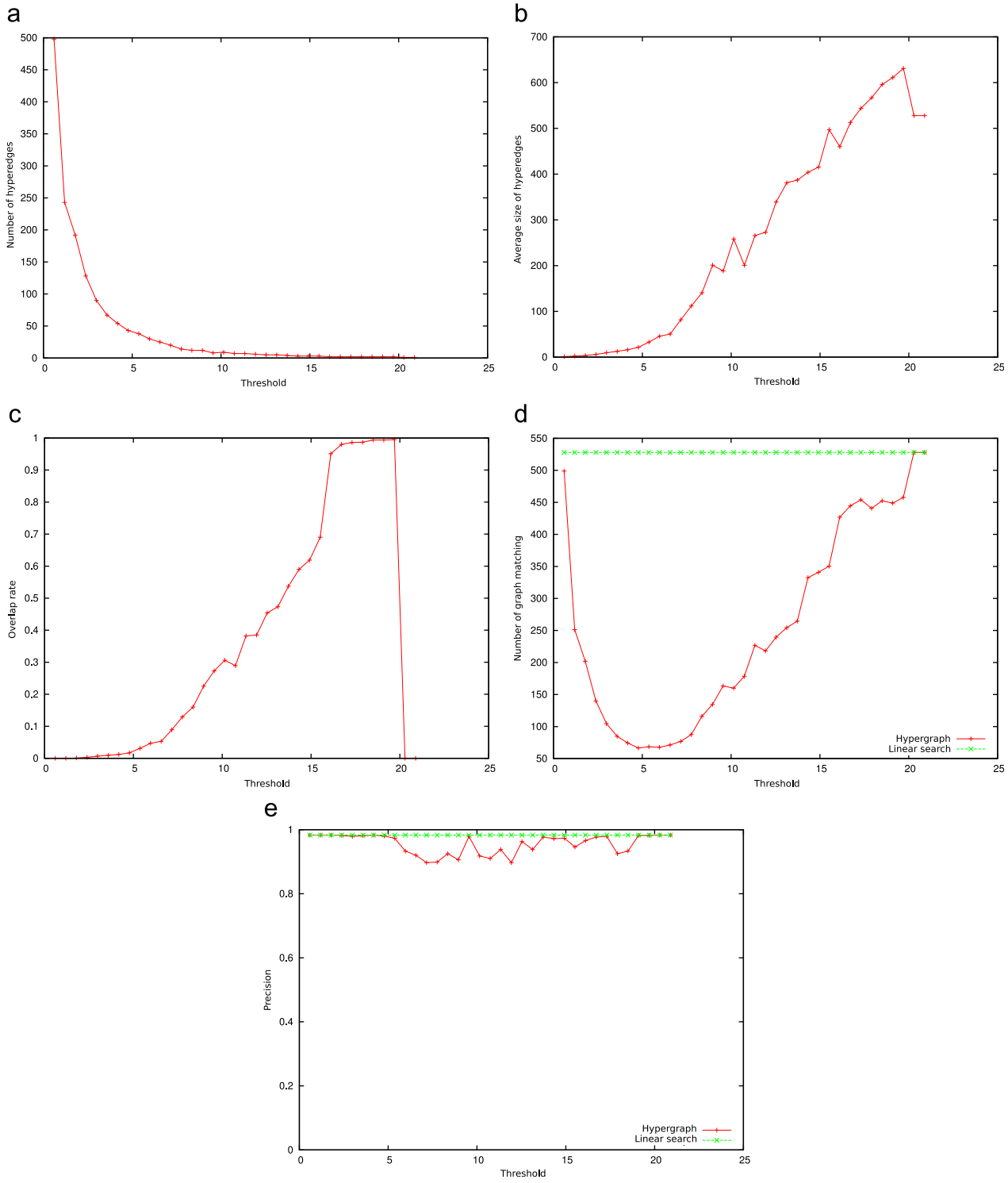

Fig. 14. Behavior of our hypergraph structure according to the threshold value, database GREC2: (a) number of hyperedges generated by our method, (b) the average size of hyperedges, (c) the overlapping rate, (d) number of graph matching and (e) retrieval precision.

Despite the fact that we explore only a portion of the database, while the classical image retrieval compares the graph of the query image with all the graphs in the database. This highlights the interest of our hypergraph model for indexing graphs. In Table 1, we illustrate a comparison between the classical retrieval method and our model by fixing the threshold value for each database. The selected threshold values correspond to the values that provide the best compromise between the maximum precision value and a minimum number of graph matching. These results show that our method shows similar performance in terms of precision than classical retrieval, although the average number of graph matching made by our method is significantly less. For example, to search for images in the Letter database, our method based on hypergraph achieves the same precision (0.90) as the classical method. On the contrary, our method performs on average 102.6 graph matching against 750 graph matching operations executed by the classical method. Similarly, we achieve similar performance for

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

Table 1
Retrieval results.

Database	Our method		Classical retrieval		Threshold
	Precision	Number of graph matching	Precision	Number of graph matching	
GREC1	0.88	69.1	0.88	528	1.25
Lettrine	0.58	87.1	0.58	280	484.98
Shape	0.53	55.3	0.53	216	4.60
Logo	0.87	29.7	0.87	80	1.44
Mutagenicity	0.64	72.8	0.64	400	2.59
Letter	0.90	102.6	0.90	750	0.16
GREC2	0.98	66.4	0.98	528	4.77

Table 2
Chemical compounds retrieval.

Data-set size	Average number of graph matching	
	Classical retrieval	Hypergraph
34,989	17,494.5	971.8

all tested databases which highlights the effectiveness and speed of our method. The effectiveness is explained by the fact that the retrieval precision of our method is equivalent to that of a linear search. Thus, the organization of graphs into a hypergraph structure does not cause a deterioration of the classical retrieval. In addition, this hypergraph structure provides a fast graph retrieval by exploring, for given retrieval task, only a subpart of the hypergraph.

5.4. Large data-set

For sake of scalability test, we propose, in this section, a retrieval process applied to a large graph set which represents chemical compounds. This data set consists of 34,989 attributed graphs. In each graph, the nodes represent the atoms labeled with the corresponding chemical symbol and edges by valence of linkage. In this experiment, only exact retrieval results will be given because no ground truth is available yet. That is, each graph is considered as a query, then the result consists of computing the number of graph matching operations performed to find the query in the data-set. We compare the number of graph matching made by our method with those of the classical retrieval in which the query is compared with all graphs in the data-set.

Table 2 presents the results of the comparison between the classical retrieval and our model by fixing the threshold value. The selected threshold value corresponds to the value that provide the best performance. These results show that our method provides very good results for large graph set comparing to classical retrieval. In addition to this scalability property and through all the experiments, we can conclude that our hypergraph model provide good results for different type of objects (graphics, documents, molecules…).

6. Conclusion

In this paper, we investigated how the structure of the hypergraph can be used for graph indexing. This organization builds a hypergraph in which each vertex is a graph and each hyperedge corresponds to a set of similar graphs. In the proposed method the organization of the set of graphs is performed in two phases: first, the method selects a set of the most representative prototypes of the set. Second, each graph is assigned to all the nearest hyperedges (created around each prototypes). The selection of the prototypes can automatically set the number of hyperedges (clusters of graphs). In addition, this also allows the multi-assignment of graphs, namely a graph can be assigned to multiple clusters. Then, our method creates an index that contains the pointers to all the selected prototypes. In the retrieval phase, only the prototypes are used to locates the set of graphs that contains the most relevant graphs to a given query. In addition, by means of a hypergraph traversal algorithm one can navigate into the graph database when the user want explore the neighborhood of a given query. The experimental results showed that our method achieves the same performance in terms of precision than classical retrieval, but the average number of graphs matched by our method is significantly less than the one provided by linear search. Besides, following the result obtained on a large chemical molecules database, our method show experimentally that it is scalable (in term of size of database) and can be useful to different domains.

Conflict of interest

None.

References

[1] S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D.J. Kriegman, S. Belongie, Beyond pairwise clustering, in: IEEE CVPR, 2005, pp. 838–845.
[2] T.R. Babu, M.N. Murty, Comparison of genetic algorithm based prototype selection schemes, Pattern Recognition 34 (2001) 523–525.
[3] R. Bayer, E.M. McCreight, Organization and maintenance of large ordered indices, Acta Informatica 1 (1972) 173–189.
[4] C. Berge, Graphes et Hypergraphes, Paris Dunod, 1970.
[5] J.C. Bezdek, M.R. Pal, J. Keller, R. Krisnapuram, Fuzzy Models and Algorithms for Pattern Recognition and Image Processing, Kluwer Academic Publishers, Norwell, MA, USA, 1999.
[6] N. Bouguila, D. Ziou, Unsupervised learning of a finite discrete mixture: applications to texture modeling and image databases summarization, Journal of Visual Communication and Image Representation 18 (2007) 295–309.
[7] H. Bunke, Recent advances in structural pattern recognition with applications to visual form analysis, in: C. Arcelli, L.P. Cordella, G.S. di Baja (Eds.), IWVF, Springer, 2001, pp. 11–23.
[8] H. Bunke, P.J. Dickinson, M. Kraetzl, M. Neuhaus, M. Stettler, Matching of hypergraphs—algorithms, applications, and experiments, in: H. Bunke, A. Kandel, M. Last (Eds.), Applied Pattern Recognition. Studies in Computational Intelligence, vol. 91, Springer, 2008, pp. 131–154.
[9] H. Bunke, P. Foggia, C. Guidobaldi, M. Vento, Graph clustering using the weighted minimum common supergraph, in: IAPR Workshop GbRPR 2003, Lecture Notes in Computer Science, vol. 2726, 2003, pp. 235–246.
[10] H. Bunke, K. Riesen, Graph classification based on dissimilarity space embedding, in: IAPR International Workshop, SSPR & SPR 2008, Lecture Notes in Computer Science, vol. 5342, 2008, pp. 996–1007.
[11] D. Comer, The ubiquitous b-tree, ACM Computing Surveys 11 (1979) 121–137.
[12] R. Datta, D. Joshi, J. Li, J.Z. Wang, Image retrieval: Ideas, influences, and trends of the new age, ACM Computing Surveys 40 (2008) 1–60.
[13] F.X. Dupé, L. Brun, Shape classification using a flexible graph kernel, in: X. Jiang, N. Petkov (Eds.), CAIP, Springer, 2009, pp. 705–713.
[14] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient graph-based image segmentation, International Journal of Computer Vision 59 (2004).
[15] M. Ferrer, D. Karatzas, E. Valveny, I. Bardají, H. Bunke, A generic framework for median graph computation based on a recursive embedding approach, Computer Vision and Image Understanding 115 (2011) 919–928.
[16] T. Gärtner, J.W. Lloyd, P.A. Flach, Kernels and distances for structured data, Machine Learning 57 (2004) 205–232.
[17] R. Giugno, D. Shasha, Graphgrep: A fast and universal method for querying graphs, in: International Conference on Pattern Recognition – Volume II, 2002, pp. 112–115.
[18] Z. Hong, Q. Jiang, S. Wang, Measuring overlap-rate in a hierarchical approach for color image segmentation, in: ICICIC '07: Proceedings of the Second International Conference on Innovative Computing, Information and Control, IEEE Computer Society, Washington, DC, USA, 2007, p. 554.
[19] S. Jouili, I. Mili, S. Tabbone, Attributed graph matching using local descriptions, in: J. Blanc-Talon, W. Philips, D.C. Popescu, P. Scheunders (Eds.), ACIVS, Springer, 2009, pp. 89–99.
[20] S. Jouili, S. Tabbone, Graph embedding using constant shift embedding, in: D. Ünay, Z. Çataltepe, S. Aksoy (Eds.), Recognizing Patterns in Signals, Speech, Images, and Videos—ICPR 2010 Contests, Springer, 2010, pp. 83–92.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016

[21] S. Jouili, S. Tabbone, Towards performance evaluation of graph-based representation, in: X. Jiang, M. Ferrer, A. Torsello (Eds.), GbRPR 2011, Springer-Verlag, 2011, pp. 72–81.
[22] S. Jouili, S. Tabbone, E. Valveny, Comparing graph similarity measures for graphical recognition, in: J.M. Ogier, W. Liu, J. Lladós (Eds.), GREC, Springer, 2010, pp. 37–48.
[23] H.W. Kuhn, The Hungarian method for the assignment problem, Naval Research Logistics Quarterly 2 (1955) 83–97.
[24] B. Luo, R.C. Wilson, E.R. Hancock, Spectral embedding of graphs, Pattern Recognition 36 (2003) 2213–2230.
[25] P. Ren, R.C. Wilson, E.R. Hancock, Spectral embedding of feature hypergraphs, in: IAPR Workshop S+SSPR, Lecture Notes in Computer Science, vol. 5342, 2008, pp. 308–317.
[26] K. Riesen, M. Neuhaus, H. Bunke, Graph embedding in vector spaces by means of prototype selection, in: F. Escolano, M. Vento (Eds.), GbRPR, Springer, 2007, pp. 383–393.
[27] A. Robles-Kelly, E.R. Hancock, Graph edit distance from spectral seriation, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2005) 365–378.
[28] A. Robles-Kelly, E.R. Hancock, A Riemannian approach to graph embedding, Pattern Recognition 40 (2007) 1042–1056.
[29] H. Samet, Foundations of Multidimensional and Metric Data Structures, Morgan Kaufmann, 2006.
[30] T.B. Sebastian, P.N. Klein, B.B. Kimia, Shock-based indexing into large shape databases, in: 7th European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 2352, 2002, pp. 731–746.
[31] K. Sengupta, K. Boyer, Organizing large structural modelbases, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1995) 321–332.
[32] L.G. Shapiro, R.M. Haralick, Organization of relational models for scene analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-4 (1982) 595–602.
[33] D. Sharvit, J. Chan, H. Tek, B.B. Kimia, Symmetry-based indexing of image databases, Journal of Visual Communication and Image Representation 9 (1998) 366–380.
[34] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain, Content-based image retrieval at the end of the early years, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 1349–1380.
[35] H. Spath, Cluster Analysis Algorithms for Data Reduction and Classification of Objects, Ellis Horwood Limited, West Sussex, 1980.
[36] S. Srinivasa, S. Kumar, A platform based on the multi-dimensional data model for analysis of bio-molecular structures, in: VLDB, 2003, pp. 975–986.
[37] A. Torsello, S.R. Bulò, M. Pelillo, Beyond partitions: Allowing overlapping groups in pairwise clustering, in: International Conference on Pattern Recognition, IEEE, 2008, pp. 1–4.
[38] A. Torsello, E.R. Hancock, Graph embedding using tree edit-union, Pattern Recognition 40 (2007) 1393–1405.
[39] J.H. Ward, Hierarchical grouping to optimize an objective function, Journal of the American Statistical Association 58 (1963) 236–244.
[40] X. Jiang, A. Munger, H. Bunke, On median graphs: properties, algorithms, and applications, IEEE Transactions on Pattern Analysis and Machine Intelligence 23 (2001) 1144–1151.
[41] X. Yan, P.S. Yu, J. Han, Graph indexing: a frequent structure-based approach, in: SIGMOD '04: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data, ACM, 2004, pp. 335–346.

Salim Jouili is a R&D architect at EURA NOVA, Belgium. He received his Master degree in computer science in 2007 from the University of Grenobe, France. In 2011, he received his Ph.D. degree in computer science at the University of Nancy 2, France, he was also a member of the QGAR research project at the LORIA-INRIA Research Center (France). His main research interests are syntactic and structural pattern recognition, indexing large-scale database, data mining, social networks, machine learning, and content-based image retrieval.

Salvatore Tabbone is a professor in computer science at the University of Nancy, France. He received his Ph.D. degree in Computer Science from Institut Polytechnique de Lorraine, France, in 1994. Since 2007, he leads the QGAR research project at the LORIA Research Center. His research topics include image and graphics document processing and indexing, pattern recognition, image filtering and segmentation, and content-based image retrieval. He has been and is the leader of several national and international projects funded by French and European institutes. He is (co-)author of more than 100 articles into refereed journals and conferences (see http://www.loria.fr/~tabbone). He serves as a program committee member for numerous national and international conferences.

Please cite this article as: S. Jouili, S. Tabbone, Hypergraph-based image retrieval for graph-based representation, Pattern Recognition (2012), http://dx.doi.org/10.1016/j.patcog.2012.04.016