A hypergraph-based model for graph clustering: application to image indexing

Salim Jouili, Salvatore Tabbone

A hypergraph-based model for graph clustering: application to image indexing

Salim Jouili and Salvatore Tabbone

LORIA UMR 7503 - University of Nancy 2
BP 239, 54506 Vandoeuvre-lès-Nancy Cedex, France
{salim.jouili,tabbone}@loria.fr

Abstract. In this paper, we introduce a prototype-based clustering algorithm dealing with graphs. We propose a hypergraph-based model for graph data sets by allowing clusters 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 define a retrieval technique indexing the database with hyperedge centroids. This model is interesting to travel the data set and efficient to cluster and retrieve graphs.

1 Introduction

Graphs give a universal and flexible framework to describe the structure and the relationship between objects. They are useful in many different application domains like pattern recognition, computer vision and image analysis. For example in the context of content-based image retrieval, the user formulate a visual query. The user's target is seldom represented by a whole image which should not be processed like a one unit, because it is generally composed by a set of visual regions carrying out some semantics. Then, the graphs, by their natures, propose an adjusted solution for this task. Moreover, to reduce the number of graphs to be computed for matching or indexing tasks it is generally required to cluster objects. By this way, clustering similar images becomes equivalent to look for those graph representations that are similar to each other in a database. In this context, it is natural to apply clustering techniques to graphs. Clustering large set of graphs is still widely unexplored and is one of the most challenging problems in structural pattern recognition. In the recent years, some investigations on graph clustering and the organization of graph databases have been revitalized in [6, 10, 13, 22]. Graph clustering problems rely in the organization of large structural databases, in discovering shape categories and view structure of objects, or in the construction of nearest neighbor classifiers. In this perspective, we propose a hypergraph model to cluster a set of graphs. A hypergraph [3] H=(ϑ, ξ) consists of a set of vertices ϑ and a set of hyperedges ξ; each hyperedge is a subset of vertices. We can note that the difference between an edge in a graph and a hyperedge in a hypergraph is that the former is always a subset of

one or two vertices, and in the latter, the subset of vertices can be of arbitrary cardinality. In our model, we represent each graph by a vertex and each cluster by a hyperedge. The *degree* of a vertex is the number of hyperedges it belongs to, and the degree of a hyperedge is the number of vertices it contains. We denote the maximum degree of a vertex v by $\Delta_\vartheta(v)$ and the maximum degree of a hyperedge h by $\Delta_\xi(h)$. Recently, the hypergraph has been used, in the pattern recognition domain, for object representation [15], similarity measures [5], and object clustering [1]. In this paper we establish a hypergraph-based model for a graph database, we process as follows: firstly, a clustering technique based on the prototype selection is proposed to cluster the graph set into k independent clusters (k is detected automatically using a given threshold). Secondly, these clusters will be overlapped to define the final hypergraph structure. The idea of clusters overlapping is in the same vein as the works in [4, 24] but the representation is different here. In fact from a set of experiments, we have remarked that the hypergraph structure provides a framework to retrieve and to browse graphs. This also leads to high clustering rate and improves the retrieval performance.

2   The proposed Hypergraph model

Since, we have focus our work in that one graph can belongs to several clusters, 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 a selection of the most representative graphs in 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, 17, 23]. K. Riesen and al. [17] 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 [23] 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 [23] 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 algorithm and the K-Centers. 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.

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. We introduce an improvement for the Leader algorithm [23]. The proposed algorithm proceeds as follows:

1. Select the median graph [11] 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 $g_i \in S \setminus \{G_{p_k}\}$ 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 predefined 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.

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. Futhermore, 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 [25] criterion:

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

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. However, we know that the distance between two centroids is higher than the threshold T and $I_i \leq T$. Moreover, by fixing the initial selected prototype as the furthest graph to the median graph of the graph set, multiple runs of the algorithm produce identical results. We denote this clustering algorithm by D-hypergraph (disconnected hypergraph).

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 $g \in S \setminus P$ 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 [18–21], but it will be difficult to find an algorithm for browsing the whole set of graphs through disjoint clusters.

On the contrary, we propose a hypergraph-based model which allows the overlapping of clusters. In fact, henceforth the clusters will be viewed as hyperedges of hypergraph and the graphs as the vertices. Firstly, for each selected prototype p_i a hyperedge h_i is defined with a centroid p_i. Secondly, every hyperedge is defined as follows : each graph $g \in S \setminus P$ 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). We denote this procedure by C-hypergraph (connected hypergraph).

Figure 1(a) illustrates our motivation. In the leftmost part of the figure $d_i = d(p_i, g_1)$ and we suppose that d_1 and d_2 are less or equal than T, so the graph g_1 shares some informations with p_1 and p_2 (informations are illustrated in colors). With the hypergraph model we will able to assign g_1 to the both hyperedges h_1 and h_2. The rightmost part of the figure 1(a) describes how two hyperedges (clusters) can overlap with one graph in common. Here, $\Delta_\vartheta(g_1) = 2$ and $\Delta_\xi(h_1) = \Delta_\xi(h_2) = 2$.
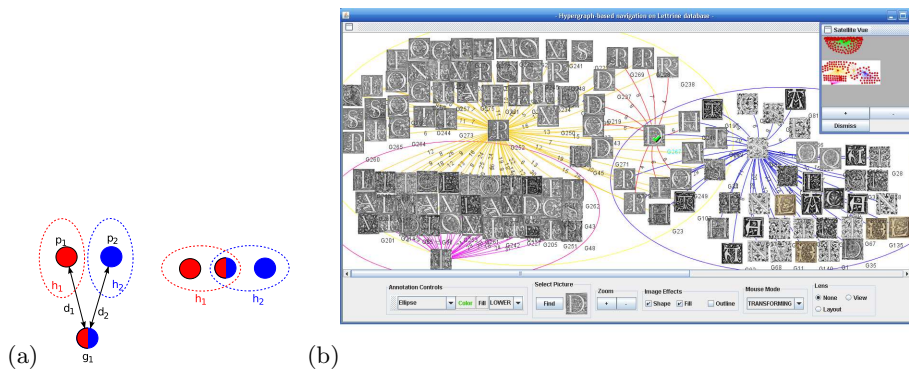

Fig. 1. Illustration of the hypergraph-based representation

Once all the hyperedges are defined from the graphs, we recompute, for each hyperedge, the generalized median graph which will be the new hyperedge centroids. The aim of this step is to update the hyperedge centroid after the hypergraph construction step and to maintain it including as much information as possible of the graphs in the corresponding hyperedge. We have chosen to use the generalized median graph to define the centroid of a cluster (unlike the Minimum Common Supergraph [6]) because it is less expensive in a viewpoint of computational time.

Interrogation and Navigation of hypergraph-based model of a set of graphs. Classically, interrogation of a set of graphs consists in searching the most similar graphs to a given query. This retrieval task ranks the graphs in an increasing distance order from the query. As remarked by a few works in the literature [18–21], this method do not exploit sophistically the distances, and the authors propose a clustering-based retrieval technique to improve the retrieval

results. Here, we introduce a procedure which involves the hypergraph-based model presented previously. The main idea is to find the most similar hyperedge centroid to a given graph 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. Get the nearest hyperedge centroid p_i to g_q.
3. Retrieve the most similar graphs g_j to g_q, where $g_j \in h_i$ and h_i is the hyperedge with the centroid p_i.

This hypergraph-based model can be exploited to travel through the hypergraph. Once the previous retrieval procedure is performed, the user can browse the set of graphs, through a graphical interface (see figure 1(b), where the clusters (hyperedges) are represented by overlapped ellipses), by performing a walk among the hyperedges.

3 Experiments

The clustering evaluation. In this first part of the experiments, our contribution is evaluated in a graph clustering context. Here, our contribution is used within two algorithms. The first one is the prototype-based clustering without connection of the hyperedges in the hypergraph (denoted D-Hypergraph as disconnected hypergraph). The second one allows the overlapping of the hyperedges (denoted C-Hypergraph as connected hypergraph). We drawn a comparison within a K-means algorithm. To this aim we have used three image databases, the first one is the well-known COIL database [14] which contains different views of 3D objects. The images in COIL are converted into graphs by feature points extraction using the Harris interest points [9] and Delaunay triangulation. The second is the well-known GREC [16, 7] database which consists of graphs representing symbols from architectural and electronic drawings. Here the ending points (ie corners, intersections and circles) are represented by nodes which are connected by undirected edges and labeled as lines or arcs. Finally, we have performed the clustering evaluation on an ornamental letters data set which contains lettrine (graphical object) extracted from digitized ancient documents [1]. Since one lettrine contains a lot of information (i.e. texture, decorated background, letters), the graphs are extracted from a region-based segmentation [8] of the lettrine. The nodes of the graph are represented by the regions and the edges describe their adjacency relationships. The graph distance measure used on the clustering is the graph matching measure based on the node signatures [12]. The clustering results are evaluated by the *Rand index*, the *Dunn index* and the *Davies-Bouldin index*. The *Rand index* measures how closely the clusters created by the clustering algorithm match the ground truth. The *Dunn index* is a

---

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

measure of the compactness and separation of the clusters and unlike the *Rand index*, the *Dunn index* is not normalized. The *Davies-Bouldin index* is a function of intra-cluster compactness and inter-cluster separation. We note that a good clustering provides a smaller *Davies-Bouldin index* and a higher *Rand* and *Dunn indexes*. In this experiment the threshold T, used by our method, is defined as the mean of distances between graphs in the same database. The number of classes k used by the K-means is defined in accordance with the ground truth.

Table 1 shows the results of the three cluster validity indexes. From these results, it is clear that our disconnected hypergraph produces clusters more compact and well separated. We note that when the C-Hypergraph is performed the *Dunn index* take the value 0, because some graphs share clusters and the minimum between-class distance becomes 0. Moreover, in a viewpoint of similarity to the ground truth, our model provides better results for the GREC and the Lettrine database, and we can remark also that the *Rand* index of the C-Hypergraph for the three databases are higher than the *Rand* index of the D-Hypergraph. Therefore, the connected hypergraph fits better the ground truth and encourages us to exploit the hypergraph-based structure for the graph retrieval problem.

	K-means	**D-Hypergraph**	**C-Hypergraph**
COIL Database	k=100	T=18.66, Nc=276	T=18.66
Rand Index	0.75	0.74	**0.75**
Dunn Index	0.03	**0.04**	0.00
DB Index	0.98	**0.88**	1.25
GREC Database	k=22	T=6.20, Nc=21	T=6.20
Rand Index	0.86	0.88	**0.91**
Dunn Index	0.01	**0.04**	0.00
DB Index	0.83	**0.76**	0.94
Lettrine Database	k=4	T=53.20, Nc=4	T=53.20
Rand Index	0.64	0.68	**0.69**
Dunn Index	0.10	**0.13**	0.00
DB Index	0.81	**0.61**	0.92

Table 1. Clustering evaluation and comparison with K-means (Nc: the number of detected clusters)

Evaluation of the retrieval with the hypergraph-based model. In this part of the experiments, we investigate the retrieval in the hypergraphs by performing the algorithm detailed previously on the Ornamental letters database. We provide a comparison with a classical retrieval task in which the graph query is compared to all the graphs in the database and then the most similar (the nearest ones) are retrieved. In the proposed approach, the hyperedges centroids are the entries of the database. That is to say, firstly the query graph is compared only to the hyperedge centroids. Then, the retrieval is performed among the graphs which belong to the hyperedge with the nearest centroid to the query. We used the receiver-operating curve (ROC) to measure retrieval performances. The ROC curves are formed by Precision rate against Recall rate, and drawn

in the figure 2. By analyzing the two curves, we can remark that the results are better when the retrieval is performed only in one hyperedge. Furthermore, the hypergraph-based model is less time-consuming than the classic technique since it does not compare the query with all graphs in the set but only with graphs in an appropriate clusters.
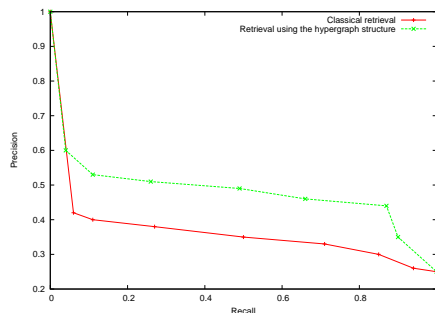

Fig. 2. Precision-Recall curves: comparison between classical retrieval and hypergraph-based retrieval

4   Conclusion

In this paper we have investigated how the hypergraph structure can be used for the purpose of graph database representation. We have proposed a prototype-based method to cluster graphs and to select automatically the prototypes which collect as much information as possible from the graph set without a predefined number of clusters. The major task of this work is to allow the multi-assignment of one graph, i.e. one graph can be assigned to more than one cluster. We have also shown that our hypergraph-based model improve the retrieval and can be used to navigate into a graph database.

References

1. S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. J. Kriegman, and S. Belongie. Beyond pairwise clustering. In *IEEE CVPR*, pages 838–845, 2005.
2. T. R. Babu and M. N. Murty. Comparaison of genetic algorithm based prototype selection schemes. *Pattern Recognition*, 34:523–525, 2001.
3. C. Berge. *Graphes et Hypergraphes*. Paris Dunod, 1970.
4. J. C. Bezdek, M. R. Pal, J. Keller, and R. Krisnapuram. *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Kluwer Academic Publishers, Norwell, MA, USA, 1999.

5. H. Bunke, P. J. Dickinson, M. Kraetzl, M. Neuhaus, and M. Stettler. Matching of hypergraphs - algorithms, applications, and experiments. In H. Bunke, A. Kandel, and M. Last, editors, *Applied Pattern Recognition*, volume 91 of *Studies in Computational Intelligence*, pages 131–154. Springer, 2008.

6. H. Bunke, P. Foggia, C. Guidobaldi, and M. Vento. Graph clustering using the weighted minimum common supergraph. *IAPR Workshop GbRPR 2003, LNCS 2726*, pages 235–246, 2003.

7. P. Dosch and E. Valveny. Report on the second symbol recognition contest. In *IAPR Workshop on GREC, LNCS 3926*, pages 381–397, 2005.

8. P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

9. C. Harris and M. Stephens. A combined corner and edge detection. In *Proceedings of The Fourth Alvey Vision Conference*, pages 147–151, 1988.

10. A. Hlaoui and S. Wang. A graph clustering algorithm with applications to content-based image retrieval. *ICMLC, 2003*, 3:1855–1861, 2003.

11. X. Jiang, A. Munger, and H. Bunke. On median graphs:properties, algorithms, and applications. *IEEE TPAMI*, 23(10):1144–1151, 2001.

12. S. Jouili and S. Tabbone. Graph matching using node signatures. In *IAPR Workshop on GbRPR, LNCS 5534*, pages 154–163, 2009.

13. B. Luo, R. C. Wilson, and E. R. Hancock. Spectral feature vectors for graph clustering. *IAPR Workshop on S+SSPR, LNCS 2396*, pages 83–93, 2002.

14. S. Nene, S. Nayar, and H. Murase. Columbia object image library (coil-100). *technical report, Columbia Univ.*, 1996.

15. P. Ren, R. C. Wilson, and E. R. Hancock. Spectral embedding of feature hypergraphs. In *IAPR Workshop S+SSPR, LNCS 5342*, pages 308–317, 2008.

16. K. Riesen and H. Bunke. Iam graph database repository for graph based pattern recognition and machine learning. In *IAPR Workshop on S+SSPR, LNCS 5342*, pages 287–297, 2008.

17. K. Riesen, M. Neuhaus, and H. Bunke. Graph embedding in vector spaces by means of prototype selection. *IAPR Workshop on GbRPR, LNCS 4538*, pages 383–393, 2007.

18. A. Robles-Kelly and E. R. Hancock. Graph edit distance from spectral seriation. *IEEE TPAMI*, 27(3):365–378, 2005.

19. T. B. Sebastian, P. N. Klein, and B. B. Kimia. Shock-based indexing into large shape databases. In *7th European Conference on Computer Vision, LNCS 2352*, pages 731–746, 2002.

20. K. Sengupta and K. Boyer. Organizing large structural modelbases. *IEEE TPAMI*, 17(4):321–332, Apr 1995.

21. L. G. Shapiro and R. M. Haralick. Organization of relational models for scene analysis. *IEEE TPAMI*, PAMI-4(6):595–602, Nov. 1982.

22. A. Shokoufandeh and S. J. Dickinson. A unified framework for indexing and matching hierarchical shape structures. In *4th Int. Workshop on Visual Form, LNCS 2059*, pages 67–84, 2001.

23. H. Spath. Cluster analysis algorithms for data reduction and classification of objects. *Ellis Horwood Limited, West Sussex*, 1980.

24. A. Torsello, S. R. Bulò, and M. Pelillo. Beyond partitions: Allowing overlapping groups in pairwise clustering. In *ICPR*, pages 1–4. IEEE, 2008.

25. J. H. Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963.