Graph-based k-means Clustering: A Comparison of the Set Median versus the Generalized Median Graph

M. Ferrer[1], E. Valveny[2], F. Serratosa[3], I. Bardají[1], and H. Bunke[4]

[1] Institut de Robòtica i Informàtica Industrial, UPC-CSIC
C. Llorens Artigas 4-6, 08028 Barcelona, Spain
{mferrer,ibardaji}@iri.upc.edu
[2] Centre de Visió per Computador, Universitat Autònoma de Barcelona
Edifici O Campus UAB, 08193 Bellaterra, Spain
ernest@cvc.uab.cat
[3] Departament d'Informàtica i Matemàtiques, Universitat Rovira i Virgili
Av. Països Catalans 26, 43007 Tarragona, Spain
francesc.serratosa@urv.cat
[4] Institute of Computer Science and Applied Mathematics, University of Bern
Neubrückstrasse 10, CH-3012 Bern, Switzerland
bunke@iam.unibe.ch

Abstract. In this paper we propose the application of the generalized median graph in a graph-based k-means clustering algorithm. In the graph-based k-means algorithm, the centers of the clusters have been traditionally represented using the set median graph. We propose an approximate method for the generalized median graph computation that allows to use it to represent the centers of the clusters. Experiments on three databases show that using the generalized median graph as the clusters representative yields better results than the set median graph.

1 Introduction

Clustering with graphs is a well studied topic in the literature, and various approaches have been proposed up to now. The classical paradigm in those approaches is to treat the entire clustering problem as a graph, that is, each element to be clustered is represented as a node and the distance between two elements is modeled by a certain weight on the edge linking the nodes [1]. Some other recent approaches propose to perform clustering directly on graph-based data. For instance in [2], the graph edit distance and the weighted mean of a pair of graphs were used to cluster graph-based data under an extension of self-organizing maps (SOMs). In [3], the authors investigated the clustering of attributed graphs by means of Function-Described Graphs (FDGs) to obtain representatives of clusters. Trees have also been used for clustering purposes. For instance, in [4] the clustering of shock trees using the tree edit distance was introduced. Finally, the extension of the k-means clustering algorithm to graph based representations was introduced in [5].

In this later approach the set median graph [6] has been used to represent the center of each cluster. Nevertheless, the concept of the generalized median graph [6] seems to be more adequate to represent the data of each cluster. Given a set of graphs, the generalized median graph [6] is defined as the graph that has the minimum sum of distances to all graphs in the set. It can be seen as the representative of the set. Thus it has a large number of potential applications in many classical algorithms for learning, clustering and classification, usually executed in the vector domain. However, its computation is exponential both in the number of input graphs and their size [7]. A number of algorithms for the generalized median graph computation have been reported in the past [6, 8, 9], but in general they suffer from either a large complexity or are restricted to special types of graphs.

In this paper we propose, for the first time, the use of the generalized median graph as the representative of a cluster in a graph-based version of the k-means algorithm. To deal with the high time and space complexity of the median graph computation, a new approximate method based on graph embedding in vector spaces is also proposed. First, we map each graph into a vector space using an approach similar to [10]. The median of the set of vectors obtained with this mapping can be easily computed in the vector space. Then, using the two closest points in the vector space and the weighted mean of a pair of graphs [11] we obtain an approximation of the median graph as the final result.

The experiments reported in this paper focus on running the k-means algorithm using the set median and the generalized median as the cluster representatives and comparing the two approaches to each other. To this end, three different databases (two of them containing real-world data) have been used. The results are evaluated through two standard clustering performance measures (the Rand index and the Dunn index). The results show that the generalized median graph yields better results than the set median graph when it is taken as the representative of a cluster. Furthermore, our procedure potentially allows us to transfer any machine learning algorithm that uses a median from the vector to the graph domain.

The rest of this paper is organized as follows. In the next section we introduce the basic concepts used in the paper. Then in Section 3 the proposed method for the median computation is described. Section 4 reports a number of experiments and present results achieved with our method. Finally, in Section 5 we draw some conclusions.

2 Background

2.1 The Graph-Based k-means Clustering Algorithm

The k-means clustering algorithm is one of the most simple and straightforward methods for clustering data [12]. The usual way is to represent the data items as a collection of n numeric values usually arranged into a vector form in the space $\mathbb{R}^n$. Then, the *Euclidean distance* in this space and the *centroid* of a set of vectors are used to compute the mean of the data in the cluster.

A graph-based version of the classic k-means clustering algorithm has been presented in [5]. The main differences consist in the distance and the centroid computation. In the former, the graph edit distance [13] is used instead of the Euclidean distance. In the latter, in order to obtain a representative of each cluster, the set median graph (see definition below) is used instead of the centroid.

2.2 Median Graph

The median graph has been proposed as a useful tool to compute a representative of a set of graphs [6]. Let U be the set of graphs that can be constructed using a given set of labels L. Given $S = \{g_1, g_2, ..., g_n\} \subseteq U$, we can distiguish between the **set median graph** $\hat{g}$, and the **generalized median graph** $\bar{g}$ of S:

$$\hat{g} = arg \min_{g \in S} \sum_{g_i \in S} d(g, g_i), \quad \bar{g} = arg \min_{g \in U} \sum_{g_i \in S} d(g, g_i)$$

where d denotes a distance or a dissimilarity measure between graphs, in our case the graph edit distance [13, 14].

~~The set median graph $\hat{g}$ is a graph g belonging to the training set S that suitably represents it. However, if we extend the search space to the whole set U, it is natural to think that a better representative (the generalized median graph) can be obtained.~~

The computation of the generalized median graph is a higly complex task, as any graph in U is a potential candidate. This makes its computation exponential in both the number and size of graphs [7]. The existing exact algorithms can only be applied to small sets of graphs with a very small number of nodes. Approximate algorithms are therefore needed [6, 9]. Thus, graph embedding techniques have been recently used to solve graph matching problems more efficiently.

引出graph embedding

2.3 Graph Embedding

Graph embedding [15] aims to convert graphs into another structure, for example, real vectors, and then operate in the associated space to make easier some typical graph-based tasks, such as matching and clustering. A first group of embedding techniques are based on spectral graph theory. For instance, a relatively early approach based on the adjacency matrix of a graph is proposed in [16]. Another similar approach has been presented in [17], where the authors use the coefficients of some symmetric polynomials constructed from the spectral features of the Laplacian matrix, to convert the graphs into a vectorial form. Finally, in a recent approach [18], the idea is to embed the nodes of a graph into a metric space and view the graph edge set as geodesics between pairs of points on a Riemannian manifold. In this work we will use another class of graph embedding procedures based on the selection of some prototypes and graph edit distance computation. This approach, which we explain in more detail in the next section, was first presented in [10], and it is based on the work proposed in [19]. The basic intuition is that the description of the regularities in observations of classes and objects is the basis to perform pattern classification. Thus, based on the selection of a number of prototypes, each object is embedded into a vector space by taking its distance to all these prototypes.

3 Median Graph via Embedding

In this section we propose a novel approach for the approximate computation of the median graph based on graph embedding in a vector space. A similar approach has been presented in [20]. Nevetheless in the present procedure, only two graphs (instead of three as in [20]) are used to recover the median graph, which simplifies this task. This new procedure consists of three steps.

In a first step, graphs are embedded into a vector space using a variation of the novel approach proposed in [10]. In that work, a set T of prototypes is used to embed each graph in a vector space. In our case, the set of prototypes is exactly the same set $S = \{g_1, g_2, ..., g_n\}$ of training graphs that are used to compute the median graph. We therefore compute the graph edit distance between every pair of graphs in the set S. Since computing the graph edit distance is a NP-complete problem, in this work we have used the suboptimal methods presented in [21, 22]. The resulting distances are arranged in a distance matrix. Each row (column) of the matrix can be seen as an n-dimensional vector. Since each row (column) of the distance matrix is assigned to one graph, such an n-dimensional vector is the vectorial representation of the corresponding graph.

Once all the graphs have been embedded in the vector space, the median vector is computed. To this end we use the concept of *Euclidean Median*. Given a set X, the *Euclidean Median* is a point $y \in \mathbb{R}^n$ that minimizes the sum of the Euclidean distances to all the points in the set. The Euclidean median has been chosen as the representative in the vector domain for two reasons. The first reason is that the median of a set of objects is one of the most promising ways to obtain the representative of such a set. The second is that, since the median graph is defined in a way very close to the median vector, we expect the median vector to represent accurately the vectorial representation of the median graph, and then, from the median vector to obtain a good approximation of the median graph. In this work we have used the most common approximate algorithm for the computation of the Euclidean median, that is, the Weiszfeld's algorithm [23].

Finally, in order to obtain the median graph, the last step is to transform the Euclidean median into a graph. Such a graph will be considered as an approximation of the median graph of the set S. To this end we will use a procedure based on the weighted mean of a pair of graphs [11].

The *weighed mean* of two graphs g and g' is a graph g'' such that $d(g, g'') = a$, $d(g'', g') = b$ and $d(g, g') = a + b$ for any two constants a and b with $0 \leq a, b \leq d(g, g')$. That is, g'' is a graph in between the graphs g and g' along the edit path between them. Figure 1 illustrates this idea.

To transform the median vector obtained in step 2 into a graph, we propose a strategy that uses two points in the vector space. The idea is the following (see Figure 2). Once the median vector v_m is computed, we choose its two closest points (v_1 and v_2 in Figure 2). Then, we compute the median vector of these two points obtaining v'_m. This point v'_m is used to obtain the approximate median graph. To this end, we first compute the distance of v_1 and v_2 to v'_m, and then, with these distances we apply the weighted mean of a pair of graphs,

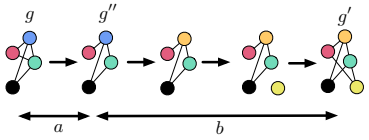

Fig. 1. Example of the weighted mean of a pair of graphs

between g_1 and g_2 (which correspond to v_1 and v_2 respectively), to obtain g'_m, the approximate median graph.

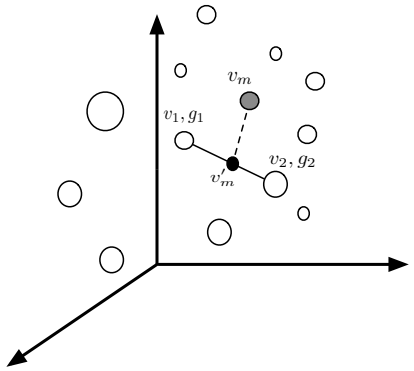

Fig. 2. Illustration of the two-point based procedure.

4 Application to Graph-Based *k*-means Clustering

In this section we propose to use the approximate method for the median graph computation to obtain the representatives of the clusters in a graph-based k-means algorithm.

4.1 Experimental Setup

To perform the clustering experiments, we used the Molecule, the Webpage and the GREC datasets from [24]. For each dataset, the experiments consisted in computing the centers of the clusters using the set median (SM) and the generalized median (GM) with the method introduced in Section 3. The number k of clusters were set according to the number of classes in the dataset. Table 1 summarizes some basic parameters of each dataset. In order to evaluate the obtained results we performed 10 repetitions of each experiment. The clustering performance was evaluated using two standard clustering performance measures, namely the *Rand index* and the *Dunn index*.

The *Rand index* R [25] measures how closely the clusters created by the clustering algorithm match the ground truth. It produces measures with values in the interval $[0, 1]$, with 1 meaning a perfect match between the result of the clustering algorithm and the ground truth.

Table 1. Number of classes and number of elements per class for each database.

Dataset	#Classes	Elements/Class
Molecules	2	100
Webpages	6	30
GREC	32	20

The *Dunn index D* [26] is a measure of the compactness and separation of the clusters. It is not an accuracy measure like the *Rand Index*. It is rather based on the assumption that in a "perfect" clustering, items in the same cluster should be similar (i.e. should have a small distance) and items in different clusters should be dissimilar (i.e. should have a large distance). Higher values of the *Dunn Index* indicate a better clustering. Unlike the *Rand Index*, the *Dunn Index* is not bounded in the interval $[0, 1]$ but in the interval $[0, \infty)$.

4.2 Results

The results for this experiment are summarized in Tables 2 and 3. In each table the minimum, mean and maximum values for the *Rand Index* (Table 2) and the *Dunn Index* (Table 3) for each dataset are shown. In both tables, the best results are marked in **bold** face.

Results based on the *Rand Index* show that in almost all cases the *GM* method obtains better results than the set median graph. More concretely, seven out of the nine best results in Table 2 correspond to the *GM* method. Since the *Rand Index* is a measure of how similar the clusters are to the ground truth, these overall results demonstrate the idea that the median graph is a good representative of a given set, better than the set median graph.

Table 2. Minimum, average and maximum values of the Rand index for different datasets.

	Minimum		**Average**		**Maximum**	
	SM	GM	SM	GM	SM	GM
Molecule	0.5072	**0.5545**	0.5620	**0.5952**	0.6205	**0.6860**
Webpages	0.6841	**0.8332**	0.8083	**0.8773**	0.8558	**0.9133**
GREC	**0.9410**	0.9340	0.9506	**0.9513**	**0.9602**	0.9566

Results based on the *Dunn Index* are shown in Table 3. Differently from the *Rand Index*, which is bound in between 0 and 1, the *Dunn Index* is not bounded. Thus, for the *Rand Index* it is relatively easy to interpret the value, because 0 means a completely uncorrelated result with respect to the groundtruth and 1 means a perfect match between the result and the groundtruth independently of the dataset used. However, the same reasoning is not possible for the *Dunn Index*. That is, we cannot say how good a result x for the *Dunn Index* is unless the *Dunn Index* for the groundtruth is given. For this reason, we have also computed the *Dunn Index* for the groundtruth (GT).

The results for each method are shown in Table 3. In this case the majority of the best results correspond to the set median. At first glance, these results could be interpreted in the sense that the set median reflects better the ideal

cluster. Actually, however, they show that the set median graph obtains a better separation of the data into compact clusters. Yet, the results of the *Dunn Index* for the groundtruth show very low values. That means that the original datasets have low separability and compactness. In this sense, the *GM* method has more similar results to the *GT* than the set median. That means that it is able to better capture the original information of the clusters.

Table 3. Minimum, average and maximum values of the Dunn index for different datasets.

	Minimum		**Average**		**Maximum**		
	SM	GM	SM	GM	SM	GM	GT
Molecule	0.0113	**0.0272**	**0.034**	0.0288	**0.0909**	0.0431	0.0182
Webpages	**0.2039**	0.1028	**0.2448**	0.2027	**0.6046**	0.5784	0.1835
GREC	0.0411	**0.0423**	0.0503	**0.0507**	**0.0651**	0.0569	0.0619

5 Conclusions

In this paper we have presented, for the first time, the use of the generalized median graph to obtain the centers of the clusters in a graph-based k-means algorithm using real-world data. To deal with the high computational requirements of the median graph computation, a new approximate method based on graph embedding in vector spaces has also been presented.

We performed a series of clustering experiments using three different databases. To evaluate the results, two standard clustering performance measures, namely the *Rand Index* and the *Dunn Index* have been used. Results in terms of the *Rand Index* show that with the median graph we obtain clusters closer to the groundtruth than using the set median graph. In addition, results given by the *Dunn Index* show that, although the set median graph obtains higher scores, the median graph obtains again results closer to the groundtruth.

With these results, we have shown that the median graph can be a better representative of a set of graphs. Furthermore, this new approximate procedure potentially allows the use of the median graph in other applications such as classification using real data.

Acknowledgements

This work has been supported by the Spanish research programmes Consolider Ingenio 2010 CSD2007-00018, TIN2006-15694-C02-02 and TIN2008-04998.

References

1. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: A review. ACM Comput. Surv. **31**(3) (1999) 264–323
2. Günter, S., Bunke, H.: Self-organizing map for clustering in the graph domain. Pattern Recognition Letters **23**(4) (2002) 405–417
3. Serratosa, F., Alquézar, R., Sanfeliu, A.: Synthesis of function-described graphs and clustering of attributed graphs. International Journal of Pattern Recognition and Artificial Intelligence **16**(6) (2002) 621–656
4. Luo, B., Robles-Kelly, A., Torsello, A., Wilson, R.C., Hancock, E.: Clustering shock trees. In: Proc. 3rd IAPR Workshop Graph-Based Representations in Pattern Recognition. (2001) 217–228

5. Schenker, A., Bunke, H., Last, M., Kandel, A.: Graph-Theoretic Techniques for Web Content Mining. World Scientific Publishing., USA (2005)
6. Jiang, X., Münger, A., Bunke, H.: On median graphs: Properties, algorithms, and applications. IEEE Trans. Pattern Anal. Mach. Intell. **23**(10) (2001) 1144–1151
7. Bunke, H., Münger, A., Jiang, X.: Combinatorial search versus genetic algorithms: A case study based on the generalized median graph problem. Pattern Recognition Letters **20**(11-13) (1999) 1271–1277
8. Hlaoui, A., Wang, S.: Median graph computation for graph clustering. Soft Comput. **10**(1) (2006) 47–53
9. Ferrer, M., Serratosa, F., Sanfeliu, A.: Synthesis of median spectral graph. In: Second Iberian Conference of Pattern Recognition and Image Analysis. Volume 3523 LNCS. (2005) 139–146
10. Riesen, K., Neuhaus, M., Bunke, H.: Graph embedding in vector spaces by means of prototype selection. In: 6th IAPR-TC-15 International Workshop, GbRPR 2007. Volume 4538 of Lecture Notes in Computer Science., Springer (2007) 383–393
11. Bunke, H., Günter, S.: Weighted mean of a pair of graphs. Computing **67**(3) (2001) 209–224
12. Mitchell, T.M.: Machine Learning. McGraw-Hill. (1997)
13. Bunke, H., Allerman, G.: Inexact graph matching for structural pattern recognition. Pattern Recognition Letters **1**(4) (1983) 245–253
14. Sanfeliu, A., Fu, K.: A distance measure between attributed relational graphs for pattern recognition. IEEE Transactions on Systems, Man and Cybernetics **13**(3) (May 1983) 353–362
15. Indyk, P.: Algorithmic applications of low-distortion geometric embeddings. In: IEEE Symposium on Foundations of Computer Science. (2001) 10–33
16. Luo, B., Wilson, R.C., Hancock, E.R.: Spectral embedding of graphs. Pattern Recognition **36**(10) (2003) 2213–2230
17. Wilson, R.C., Hancock, E.R., Luo, B.: Pattern vectors from algebraic graph theory. IEEE Trans. Pattern Anal. Mach. Intell. **27**(7) (2005) 1112–1124
18. Robles-Kelly, A., Hancock, E.R.: A Riemannian approach to graph embedding. Pattern Recognition **40**(3) (2007) 1042–1056
19. Pekalska, E., Duin, R.P.W., Paclík, P.: Prototype selection for dissimilarity-based classifiers. Pattern Recognition **39**(2) (2006) 189–208
20. Ferrer, M., Valveny, E., Serratosa, F., Riesen, K., Bunke, H.: An approximate algorithm for median graph computation using graph embedding. In: Proceedings of 19th ICPR. (2008) 287–297
21. Neuhaus, M., Riesen, K., Bunke, H.: Fast suboptimal algorithms for the computation of graph edit distance. In: Joint IAPR International Workshops, SSPR and SPR 2006. Lecture Notes in Computer Science 4109. (2006) 163–172
22. Riesen, K., Neuhaus, M., Bunke, H.: Bipartite graph matching for computing the edit distance of graphs. In: 6th IAPR-TC-15 International Workshop, GbRPR 2007. Volume 4538 of Lecture Notes in Computer Science., Springer (2007) 1–12
23. Weiszfeld, E.: Sur le point pour lequel la somme des distances de n points donnés est minimum. Tohoku Math. Journal (43) (1937) 355– 386
24. Riesen, K., Bunke, H.: IAM graph database repository for graph based pattern recognition and machine learning. In: SSPR/SPR. (2008) 287–297
25. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistival Association **66** (1971) 846–850
26. Dunn, J.: Well separated clusters and optimal fuzzy partitions. Journal of Cibernetics **4** (1974) 95–104