An image-based representation for graph classification

Frédéric Rayar and Seiichi Uchida

Kyushu University, Fukuoka 819-0395, Japan
{rayar,uchida}@human.ait.kyushu-u.ac.jp

Abstract. This paper proposes to study the relevance of image representations to perform graph classification. To do so, the adjacency matrix of a given graph is reordered using several matrix reordering algorithms. The resulting matrix is then converted into an image thumbnail, that is used to represent the graph. Experimentation on several chemical graph data sets and an image data set show that the proposed graph representation performs as well as the state-of-the-art methods.

Keywords: graph classification, graph representation, matrix reordering, chemoinformatics

1 Introduction

Graphs are efficient and powerful structures to represent real-world data in several fields, such as bioinformatics [5], social networks analysis [2] or pattern recognition [30]. Formally, a graph is an ordered pair $G = (V, E)$, where $V = \{v_1, ..., v_n\}$ is a set of vertices (or nodes), and $E \subset V \times V$ is a set of edges that represent relations between elements of V.

Graph classification [29] is an important and still challenging task, that has been widely addressed by the research community. This task falls into the supervised learning field, where one has to predict the label of an object that is represent by a graph. More formally, given a training set $\{g_i, l_i\}$ of graphs and their labels, one has to predict the label l of an unseen graph g. Among the many studies that have been proposed to address the graph classification problem, the most used paradigms are the graph kernels [13], along with the graph edit distance [8] (GED) for error-tolerant graph matching, and more recently graph neural networks [17]. However, these paradigms face tough challenges such as the computational requirement when performing pairwise graph comparison, which is emphasised when dealing large data sets. Regarding neural networks, despite the efforts from the research community, the adaptation of convolution and pooling operations is non-trivial for non-Euclidean objects such as graphs, and still remains a challenge.

In this paper, we propose a novel image-based representation to describe graphs, and leverage this descriptor to perform fast graph classification, while obtaining accuracies comparable with the state-of-the-art methods. The rest of the paper is organised as follows: Section 2 presents an overview of graph classification and graph visualisation paradigms. Section 3 details the proposed framework to obtain a graph's image representation. The experimentation setup is given in Section 4 and the results that have been obtained are discussed in Section 5. Finally, we conclude this study in Section 6.

2 Related works

2.1 Graph classification

Many solutions can be found in the literature to perform graph classification. These methods often boil down to compare graphs between them, and the matching can be done in either:

1. *a vector space*: in this paradigm, one aims to represent a graph in a vector space to take advantage of statistical approaches. Often referred as graph embedding, a mapping ϕ function projects the graph in $\mathbb{R}^n$:

$$\begin{aligned}\phi :& G \rightarrow \mathbb{R}^n \\ & g \mapsto \phi(g) = (f_1, ..., f_n).\end{aligned}$$

   Several approaches can be used, such as: (i) feature extraction [26] (*e.g.* number of nodes, number of edges, average degree of the nodes, number of cycles with a certain length, ...), (ii) spectral method [18] or (iii) dissimilarity representation [23] (based on distances to a set of prototype graphs).
2. *the graph space*: in this paradigm, one uses graph matching methods to compare graphs in their original space. For instance, GED [8] is a well-known error-tolerant inexact graph matching algorithm. Given a set of graph edit operations (commonly insertion, deletion, substitution), the graph edit distance between two graphs g_1 and g_2 is given by:

$$GED(g_1, g_2) = \min_{(e_1,...,e_k) \in \mathcal{P}(g_1,g_2)} \sum_{i=1}^{k} c(e_i),$$

   where $\mathcal{P}(g_1, g_2)$ is the set of edit paths to transform g_1 into g_2 and $c(e)$ is the cost of a graph edit operation e.
3. *a kernel space*: here, one leverages the kernel trick [15] to compute a similarity measure between two graphs. Kernel methods provide an implicit graph embedding and use various type of kernel, such as: random walk kernel [31], shortest-path kernel [4] or graphlet kernel [25]. One main limitation of such methods is that the extracted features are often not independent [32].

More recently, the performance of artificial neural networks has motivated their usage for graph classification. Three approaches can be considered:

1. adapting the architecture of convolutional neural networks (CNN) to deal with graph structures (*e.g.* [20]),

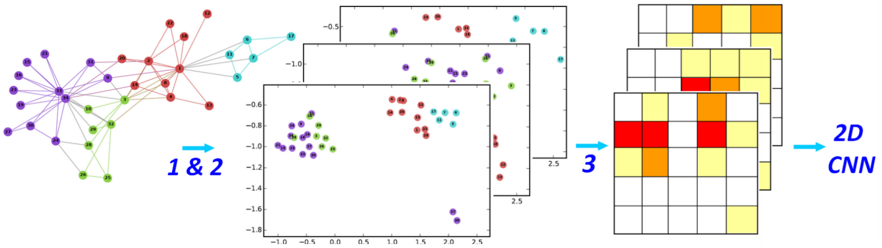


Fig. 1. Tixier et al. framework. First, a node embedding is done along with a PCA compression (1 & 2). Then, 2D histograms are extracted and stacked to build a multi-channel image-like structure (3). Illustration from the original paper [28].

2. building architecture dedicated to networks (*e.g.* [24]),
3. image-based graph representation: *i.e.* using an actual image representation along with a CNN.

This latter approach is the first motivation of this work: computing an image represention from a graph and leverage it to use a vanilla CNN. To the best of our knowledge, only one study [28], parallel to ours and recently submitted to the arXiv repository, adopts this strategy. Indeed, in [28], Tixier et al. compute *"a multi channel image-like structure to represent a graph"*. The following steps are performed: (i) graph node embedding using node2vec [14], (ii) embedding space compression using Principal Component Analysis (PCA) and (iii) computation of fixed-size 2D histograms (that will be considered as the channels of the final image-like structure). Figure 1 illustrates their proposed framework. Even if their framework achieves classification accuracies that are comparable to baseline on several data sets, the embedding of nodes is a non-trivial step, and many parameters have to be tuned (number of channel, node2vec parameters, ...).

Hence, in this study, we propose to take advantage of existing graph visualisation techniques to build a relevant image representation for graph classification, without the need of numerous parameters.

2.2 Graph visualisation

Graph drawing is a field that addresses the issue of visual depiction of graphs in two (or three) dimensional surfaces. To do so, it takes benefit of graph theory and information visualisation fields. There is two common ways to draw graphs:

- *node-link diagrams*: in such depictions, vertices of the graph are represented as disks, boxes, or textual labels. The edges are represented as segments or curves in the plane. Producing aesthetic visualisations, it is the most commonly used visualisation for graph. However, it suffers of limitations such as overlapping nodes, edge-crossing, or slow interaction for large graphs.

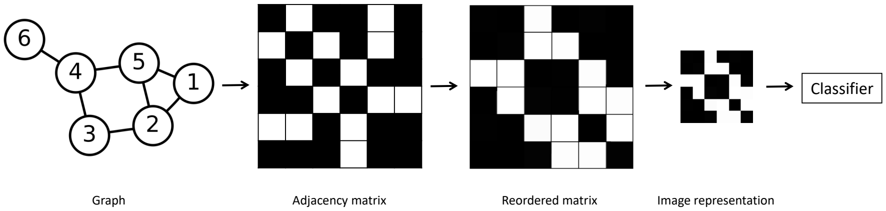


Fig. 2. Proposed framework. To represent a graph as an image, we: (i) build its adjacency matrix, (ii) apply a matrix reordering algorithm on the adjacency matrix, and (iii) convert the resulting reordered matrix into an image with predefined dimensions. This thumbnail is then given to a classifier to predict its label.

- *matrix-based visualisations*: here, the adjacency matrix of the graph is visualised. It is rarely used and most users are not familiar with this depiction, despite its *"outstanding potential"* according to [12]. Its main limitation is the fact that this visualisation is sensible to the node ordering and may produced different matrices for two graphs that have the same structure.

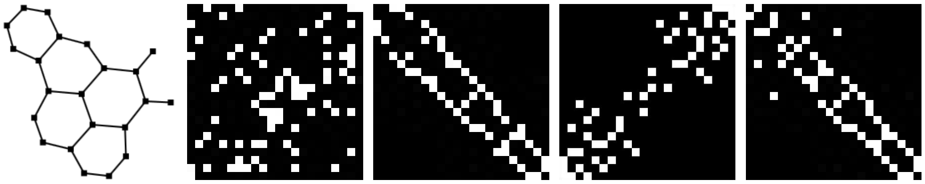
3 Proposed framework

In this study, we propose to use a matrix-based visualisation of a graph and convert it to an image. This image-based representation is then be reshaped into a vector a given to classic classifier (such as k-nearest neighbour or support vector machines (SVM)) or directly feed a CNN.

Figure 2 illustrates the proposed framework. First, the adjacency matrix is extracted from the graph. We build a binary matrix $A \in \mathfrak{M}_n$, where $a_{i,j} = 1$ if there is an edge between vertices v_i and v_j, 0 otherwise. Second, a matrix reordering algorithm is applied on the original adjacency matrix. An image version of the reordered matrix is built, and normalised to a predefined and fixed dimensions. A classic linear interpolation algorithm was used in our study. This final thumbnail is the proposed image-based representation of the graph.

The second step, that consists in applying a matrix reordering algorithm allows us to address the issue of the matrix-based visualisation node ordering sensibility. This will make the representation non-stochastic and also maintain spatial relevance in the obtained image. In this study, we investigate several approaches to reorder matrices, that have been selected according to two studies ([19] and [3]) on matrix reordering methods for graph visualisation. Indeed, the results of theses algorithms generally present perceivable and interpretable patterns, while heuristic implementations can be found in the literature to tackle their complexity. Namely, we investigate the following algorithms:

1. *minimum degree algorithm* [10] (MD): in numerical linear algebra, this algorithm is used to permute the rows and columns of a symmetric sparse matrix, before applying the *Cholesky decomposition*.



node-link MD RCM Seriation Sloan

Fig. 3. Image representations of *”4,5-dimethylbenzo[a]pyrene_sloan”* molecule appearing in the PAH data set. From left to right: a node-link diagram obtained using the Fruchterman-Reingold algorithm [7] and proposed thumbnails using minimum degree, reverse Cuthill-McKee, Seriation and Sloan matrix reordering algorithms.

2. *reverse Cuthill-McKee algorithm* (RCM): the Cuthill-McKee [6] and the reverse Cuthill-McKee [11] algorithm both aim at reducing the *bandwidth* of sparse matrices.
3. *a seriation algorithm* [16] (Seriation): introduced by specialists of archaeology and palaeontology, it boils down to finding the best enumeration order of a set of objects according to a given correlation function (*e.g.* characteristic of the data, chronological order or sequential structure within the data).
4. *Sloan algorithm* [27] (Sloan): this reordering algorithm aims at reducing the *profile* and the *wavefront* of a graph. A main advantage of this algorithm is that it takes into account both global and local criteria for the reordering process.

We refer the interested readers to [3] for a more thorough survey and details on reordering algorithms. Figure 3 illustrates the different image representations obtained using the four aforementioned matrix reordering algorithms, for a given graph.

4 Experimental setup

4.1 Data sets

Four real-world graph data sets have been used in our experimentation:

1. GREC: this data set consists of a subset of a symbol image database. It is composed of 1100 graphs, spread among 22 classes.
2. MAO: this data set is composed of 68 molecules divided into 2 classes: molecules that inhibit the monoamine oxidase (antidepressant drugs) and molecules that do not.
3. MUTA: this data set consists in $4,337$ molecules, divided in 2 classes: mutagen and nonmutagen.
4. PAH: this data set is composed of 94 molecules, also divided in 2 classes: cancerous or not cancerous molecules.

These data sets are publicly available from the IAM Graph Database Repository [22] or the GREYC's Chemistry dataset [1]. The 3 first data sets are weighted and both nodes and edges are labelled. Only the PAH data set can be viewed as unweighed and not labelled, since all atoms (nodes) are carbons and all bounds (edges) are aromatics. However, for all the four data sets, we discard the weight and the nodes/edges labels. This boils down to focusing on the structure of the graphs, and generates binary adjacency matrix (1 if there is an edge, else 0), and thus binary image representation of the graphs. This choice is justified by the fact that the present study aims at evaluating the relevance of the proposed image-based representation for graph classification. In future works, greyscale and multi-channel images will be considering to handle edge weights and node/edge labels.

4.2 Implementation

All graphs input are in .gxl format and can be viewed using the online GXL Viewer platform [2]. Regarding the algorithm, we have used the C++ boost (1.58.00) graph library [3] implementation of the minimum degree, the reverse Cuthill-McKee and the Sloan algorithms. For the Seriation algorithm, we have used the R seriation package [4].

Once the image versions of the reordered matrix are obtained, we resize them to a fixed sized of 28 × 28. This was inspired by our former goal of using CNN. Indeed, CNN performs very well on MNIST [5], an isolated handwritten digits data set, that has 28 × 28 images. We did not investigate the sensibility of the sole parameter of our approach at the present time.

Regarding the classifiers, we have used in these first experiments the 1-nearest neighbour (1-NN) and the 3-nearest-neighbour (3-NN) classifiers. Experiments have been done on both given train/test data sets for fair comparison with state-of-the-art results but also on the whole data set (with 10-fold cross-validation) for more generalised results.

5 Results and discussion

5.1 Comparison with GDC 2016

During the ICPR 2016 conference, the Graph Distance Contest (GDC 2016) [6] has been held. Two challenges have been proposed: (1) computation of the exact or an approximate graph edit distance and (2) computation of a dissimilarity measure for graph classification. Two participants have joined the second challenge, however, since the results of this challenge have not been published yet, we

[1] https://brunl01.users.greyc.fr/CHEMISTRY/index.html

[2] http://rfai.li.univ-tours.fr/PublicData/gxlviewer/

[3] https://www.boost.org/doc/libs/1_58_0/libs/graph/doc/sparse_matrix_ordering.html

[4] https://CRAN.R-project.org/package=seriation

[5] http://yann.lecun.com/exdb/mnist/

[6] https://gdc2016.greyc.fr/

Table 1. Classification results. The recognition rate (in percentage) for the four studied matrix reordering methods on the $GREC$, MAO and $MUTA$ data sets. Both 1-NN and 3-NN classifier have been used, on the train/test data sets of the GDR 2016 Challenge 2. The results obtained by the two participants of this challenge are also presented.

	#train/test	classifier	MD	RCM	Seriation	Sloan	Algo 1	Algo 2
GREC	484/528	1-NN	**91.67**	90.53	90.91	91.48	-	-
		3-NN	89.58	89.20	89.20	90.53	93.39	**99.38**
MAO	32/32	1-NN	81.25	**87.50**	75.00	81.25	-	-
		3-NN	**84.38**	**84.38**	68.75	71.88	68.75	75.00
MUTA	1800/2337	1-NN	58.54	**61.87**	60.63	61.70	-	-
		3-NN	57.60	64.18	59.35	61.45	**73.50**	48.55

do not disclose the name of the participants, and their methods will be referred as Algo 1 and Algo 2 in the rest of the paper. The organisers of the contest kindly provided us with the results of the challenge to allow us to compare our contribution in a fair context. Only the 3-NN has been used in the challenge 2.

In order to compare the relevance of the proposed image-based representation for graph classification, we used their train/valid/test partitioning of the GREC, MAO and MUTA data sets (the organisers have removed 10% on the original training data sets). Since the proposed approach do not need a validation step, the classes of the test graphs are predicted using 1-NN and 3-NN classifiers on the {train;valid} subsets.

The results of this experiment are presented in Table 1. As one can see, the proposed image-based graph representations do not allow to always outperform existing methods. However, the obtained results are comparable with the one of Algo 1 and Ago 2 and for the MAO data set, we do indeed outperform the two participant algorithm by 10%. Furthermore, unlike our proposed representations, the participants may have used the attributes of the nodes and labels during the classification process. This supports the fact that our proposed image-based representation is a relevant graph representation for graph classification.

5.2 Overall classification accuracies

In order to generalise the results, but also to present results on the PAH data set, we have conducted 10-fold cross-validation experiments. Indeed, according to the organisers of the contest [1], *"PAH represented the most challenging dataset since it is composed of large unlabelled graphs"* (all nodes are carbons and all edges are aromatics).

Table 2. Classification results (2). The recognition rate (in percentage) for the four studied matrix reordering methods on the four data sets. Both 1-NN and 3-NN have been used to perform a 10-fold cross-validation technique.

	#train/test	classifier	MD	RCM	Seriation	Sloan
GREC	990/110	1-NN	91.00	91.64	91.64	**92.45**
		3-NN	90.45	**91.18**	90.36	90.36
MAO	61/7	1-NN	79.05	**83.33**	76.19	81.90
		3-NN	**86.90**	85.24	80.95	79.52
MUTA	84/110	1-NN	62.30	**64.72**	62.35	64.26
		3-NN	59.65	**65.09**	61.59	63.15
PAH	84/110	1-NN	67.11	63.44	61.89	**72.56**
		3-NN	62.89	**70.00**	59.44	67.00

Table 2 presents the results related to this second set of experiments. We observe the same behaviour as the previous experiments: first, the accuracies are comparable to state-of-the-art methods for the three first data sets. Regarding the PAH data set, the GREYC's Chemistry dataset website mention the best classification accuracy achieved: 80.7% with the method presented in [9]. Second, we observe that using the 3 first nearest neighbours to classify unseen graphs do not always allow to increase the overall recognition accuracy. Finally, according to the results, even if MD and Sloan algorithms allow to have better recognition accuracies, we can not definitely conclude that a specific matrix reordering algorithm is best fit in our framework.

5.3 Discussion

We propose a framework where an image-based representation is leveraged to perform graph classification. The main advantage of our framework is its simplicity, that allows fast computation times while having promising accuracy results. Indeed, using greyscale or multi-channel image (without any heavy additional processes), we may considerer improving these recognition accuracies.

The major limitation of our framework, is that one does not actually compute the graph matching function, which could be a relevant asset for understanding the classification results. However, since our framework provides quickly the (dis)similarities with the training data set, one can then run a graph matching algorithm on the K first nearest neighbours in a parallel scheme, and then visualise the obtained matching with a platform such as the one proposed by [21].

6 Conclusion

The main contribution of this study is to show the feasibility of using a simple yet relevant image-based representation for graph classification. Our approach allows to obtain recognition accuracies that are comparable or better than the state-of-the-art methods, while avoiding the complexity of these methods.

These promising first results allow to consider several future works: (i) the usage of greyscale and multi-channel images, to take into account edge weights and nodes/edges labels (the latter being more challenging), (ii) the usage of a combination of images to represent a graph, or boosting technique, (iii) the usage of another classifier such as SVM or CNN, that may allow to increase the recognition accuracies. Finally, it could be interesting to apply our framework on the data sets used by Tixier et al., to compare our approaches.

Acknowledgement

The authors would like to give credits to the organisers of the Graph Distance Contest, who provided the challenge data sets and the results of the second challenge. This research was partially supported by MEXT-Japan (Grant No. 17H06100).

References

1. Abu-Aisheh, Z., Gazere, B., Bougleux, S., Ramel, J.Y., Brun, L., Raveaux, R., Hroux, P., Adam, S.: Graph edit distance contest. Pattern Recognition Letters **100**(C), 96–103 (2017)
2. Barnes, J., Harary, F.: Graph theory in network analysis. Social Networks **5**(2), 235 – 244 (1983)
3. Behrisch, M., Bach, B., Riche, N.H., Schreck, T., Fekete, J.: Matrix reordering methods for table and network visualization. Computer Graphics Forum **35**(3), 693–716 (2016)
4. Borgwardt, K.M., Kriegel, H.P.: Shortest-path kernels on graphs. In: Proceedings of the Fifth IEEE International Conference on Data Mining. pp. 74–81. IEEE Computer Society (2005)
5. Chacko, E., Ranganathan, S.: Graphs in Bioinformatics, chap. 10, pp. 191–219. Wiley-Blackwell (2010)
6. Cuthill, E., McKee, J.: Reducing the bandwidth of sparse symmetric matrices. In: Proceedings of the 1969 24th National Conference. pp. 157–172. ACM (1969)
7. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. Softw. Pract. Exper. **21**(11), 1129–1164 (1991)
8. Gao, X., Xiao, B., Tao, D., Li, X.: A survey of graph edit distance. Pattern Analysis and Applications **13**(1), 113–129 (2010)
9. Gaüzère, B., Brun, L., Villemin, D.: Graph kernel encoding substituents' relative positioning. In: International Conference on Pattern Recognition (2014)
10. George, A., Liu, J.W.: The evolution of the minimum degree ordering algorithm. SIAM Review **31**(1), 1–19 (1989)

11. George, J.A.: Computer Implementation of the Finite Element Method. Ph.D. thesis, Stanford, CA, USA (1971)
12. Ghoniem, M., Fekete, J.D., Castagliola, P.: On the readability of graphs using node-link and matrix-based representations: A controlled experiment and statistical analysis. Information Visualization **4**(2), 114–135 (2005)
13. Ghosh, S., Das, N., Gonalves, T., Quaresma, P., Kundu, M.: The journey of graph kernels through two decades. Computer Science Review **27**, 88 – 111 (2018)
14. Grover, A., Leskovec, J.: Node2vec: Scalable feature learning for networks. In: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 855–864. ACM (2016)
15. Hofmann, T., Schlkopf, B., Smola, A.J.: Kernel methods in machine learning. The Annals of Statistics **36**(3), 1171–1220 (2008)
16. Ihm, P.: A contribution to the history of seriation in archaeology. In: Classification - the Ubiquitous Challenge. pp. 307–316. Springer Berlin Heidelberg (2005)
17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
18. Luo, B., Wilson, R.C., Hancock, E.R.: Spectral embedding of graphs. Pattern Recognition **36**(10), 2213 – 2230 (2003)
19. Mueller, C., Martin, B., Lumsdaine, A.: A comparison of vertex ordering algorithms for large graph visualization. In: 2007 6th International Asia-Pacific Symposium on Visualization. pp. 141–148 (2007)
20. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. CoRR **abs/1605.05273** (2016), http://arxiv.org/abs/1605.05273
21. Rayar, F., Abu-Aisheh, Z.: Photo(Graph) Gallery: An " exhibition " of graph classification. In: International Conference on Information Visualisation (2017)
22. Riesen, K., Bunke, H.: IAM graph database repository for graph based pattern recognition and machine learning. Pattern Recognition Letters **5342**, 287–297 (2008)
23. Riesen, K., Bunke, H.: Graph Classification and Clustering Based on Vector Space Embedding. World Scientific Publishing Co., Inc. (2010)
24. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Transactions on Neural Networks **20**(1), 61–80 (2009)
25. Shervashidze, N., Vishwanathan, S., Petri, T., Mehlhorn, K., Borgwardt, K.: Efficient graphlet kernels for large graph comparison. In: International Conference on Artificial Intelligence and Statistics. vol. 5, pp. 488–495. PMLR (2009)
26. Sidere, N., Heroux, P., Ramel, J.Y.: A vectorial representation for the indexation of structural informations. In: Structural, Syntactic, and Statistical Pattern Recognition. pp. 45–54. Springer Berlin Heidelberg (2008)
27. Sloan, S.W.: An algorithm for profile and wavefront reduction of sparse matrices. International Journal for Numerical Methods in Engineering **23**(2), 239–251 (1986)
28. Tixier, A.J., Nikolentzos, G., Meladianos, P., Vazirgiannis, M.: Classifying graphs as images with convolutional neural networks. CoRR **abs/1708.02218** (2017), http://arxiv.org/abs/1708.02218
29. Tsuda, K., Saigo, H.: Graph Classification, pp. 337–363. Springer US (2010)
30. Vento, M.: A long trip in the charming world of graphs for pattern recognition. Pattern Recognition **48**(2), 291 – 301 (2015)
31. Vishwanathan, S.V.N., Borgwardt, K.M., Schraudolph, N.N.: Fast computation of graph kernels. In: Proceedings of the 19th International Conference on Neural Information Processing Systems. pp. 1449–1456. MIT Press (2006)
32. Yanardag, P., Vishwanathan, S.V.N.: Deep graph kernels. In: KDD (2015)