2011 International Conference on Document Analysis and Recognition

Subgraph Spotting through Explicit Graph Embedding: An Application to Content Spotting in Graphic Document Images

Muhammad Muzzamil Luqman*†, Jean-Yves Ramel*, Josep Lladós† and Thierry Brouard*
*Laboratoire d'Informatique, Université François Rabelais de Tours, 37200 France.
†Computer Vision Center, Universitat Autònoma de Barcelona, 08193 Spain.
Email: {mluqman,josep}@cvc.uab.es, {ramel,brouard}@univ-tours.fr

***Abstract*—We present a method for spotting a subgraph in a graph repository. Subgraph spotting is a very interesting research problem for various application domains where the use of a relational data structure is mandatory. Our proposed method accomplishes subgraph spotting through graph embedding. We achieve automatic indexation of a graph repository during off-line learning phase; where we (i) break the graphs into 2-node subgraphs (a.k.a. cliques of order 2), which are primitive building-blocks of a graph, (ii) embed the 2-node subgraphs into feature vectors by employing our recently proposed explicit graph embedding technique, (iii) cluster the feature vectors in classes by employing a classic agglomerative clustering technique, (iv) build an index for the graph repository and (v) learn a Bayesian network classifier. The subgraph spotting is achieved during the on-line querying phase; where we (i) break the query graph into 2-node subgraphs, (ii) embed them into feature vectors, (iii) employ the Bayesian network classifier for classifying the query 2-node subgraphs and (iv) retrieve the respective graphs by looking-up in the index of the graph repository. The graphs containing all query 2-node subgraphs form the set of result graphs for the query. Finally, we employ the adjacency matrix of each result graph alongwith a score function, for spotting the query graph in it. The proposed subgraph spotting method is equally applicable to a wide range of domains; offering ease of query by example (QBE) and granularity of focused retrieval. Experimental results are presented for graphs generated from two repositories of electronic and architectural document images.**

***Keywords*: subgraph spotting, explicit graph embedding, graphics recognition, content spotting, focused retrieval.**

I. Introduction and related works

In last few years, content based information retrieval (CBIR) systems for graphic document image repositories, have emerged as an important application domain. This has resulted into a gradual shift of attention from the hard problems of symbol recognition and localization to the relatively softer problem of content spotting (a.k.a. symbol spotting). This is a direct outcome of growing size of document image repositories and the increasing demand from users to have an efficient browsing mechanism for graphic content. The format of these documents restricts the use of classical keyword based indexing mechanisms. Thus a very interesting research problem is to investigate into mechanisms of automatically indexing the content of graphic document images; in order to offer to users the ease of query by example (QBE) and granularity of focused retrieval.

Graphs have remained a very popular choice of graphics recognition research community [1][2]. These data structures do not suffer from fixed dimensionality. They are able to represent both symbolic and numeric properties of an object and can explicitly model the relations between parts of an object. The graph based structural representations are very convenient and powerful for relational information, and have their application to a wide range of domains [3][4]. Keeping in view the wide use of graph based representations for graphic document images, the problem of content spotting in graphic document image repositories, in fact, becomes the problem of subgraph spotting.

Important recent works on content based graphic document image retrieval include [5][6]. In this paper we take forward our work on content spotting in graph representation of line-drawing graphic document images. In [7] we proposed a content spotting system for line-drawing graphic document images, for taking the keyword based information retrieval for them one step forward to query by example (QBE) and focused retrieval. The system comprises of an off-line unsupervised learning phase and an on-line content spotting phase. During off-line (unsupervised learning) phase, first it vectorizes the document images and represents them by attributed relational graphs. The work of Qureshi et al. [8] has been employed for extracting the basic building blocks of the graphic document images - Qureshi et al. [8] have proposed a set of heuristics for localizing symbols or (more generally speaking) the regions of interest (ROIs) in attributed relational graph representation of line-drawing architectural and electronic document images. The ROIs are represented by subgraphs and are embedded into numeric feature vectors which are termed as fuzzy structural signatures. The fuzzy structural signatures are clustered into classes using a classic agglomerative clustering technique. The off-line learning phase completes by building an index for the document image repository (i.e. document image vs ROI vs cluster-id) and learning a Bayesian network classifier. During the on-line content spotting phase the user selects a region in a document image for posing a query to the system. The on-line content spotting phase follows

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.178

the same sequence of steps (as outlined for unsupervised learning phase) for embedding the query image ROIs into fuzzy structural signature and employs a Bayesian network classifier for recognizing the ROIs in query image. It finally retrieves the corresponding documents by looking-up in the repository index and focuses on the query ROIs in the retrieved document images.

Our novel proposed method generalizes our work in [7], to subgraph spotting and facilitates the smooth deployment of our system to a wide range of application domains. Instead of relying on domain specific heuristics, we propose to use a more primitive building-block for indexation of graph repositories. A clique of order 2 or a 2-node subgraph is actually the most primitive building-block of a graph. A very important problem which arises as a consequence of the choice of using such a basic subgraph is the extraction of useful information for efficiently distinguishing two 2-node subgraphs. We achieve this by employing our recently proposed explicit graph embedding technique for embedding attributed graphs into feature vectors [9]. Apart from incorporating learning abilities in structural representations (without requiring any labeled training set) and offering the ease of query by example (QBE) and the granularity of focused retrieval, our proposed method does not impose any restriction on the size of query subgraph, as far as it contains at-least one 2-node subgraph.

The proposed system offers an improvement to indexation part of off-line unsupervised learning phase of [7] and gives a very general solution for subgraph spotting; indeed equally applicable to a whole range of domains where the use of a relational data structure is mandatory. The fact that our method does not require any labeled training set enables its less expensive and fast deployment to various domains. A second, worth highlighting, contribution of this work is the score function for localizing a subgraph in a graph.

In Section II we introduce definitions and notations used in this paper. Section III outlines our proposed method of subgraph spotting. Experimental results for graphic document image repositories are presented in Section IV and the paper concludes in Section V with future directions of the work.

II. Definitions and notations

Attributed Graph (AG): Let A_V and A_E denote the domains of possible values for attributed vertices and edges. These domains are assumed to include a special value that represents a null value of a vertex or an edge. An attributed graph AG over (A_V, A_E) is defined to be a four-tuple:

$$AG = (V, E, \mu^V, \mu^E)$$

where,
V is a set of vertices,
$E \subseteq V \times V$ is a set of edges,
$\mu^V : V \longrightarrow A_V$ is function assigning attributes to vertices and
$\mu^E : E \longrightarrow A_E$ is a function assigning attributes to edges.

In an attributed graph AG: $|V|$ is graph order, $|E|$ is graph size and degree of a node is number of edges connected to it.

Graph Embedding: Graph Embedding is a methodology aimed at representing a whole graph (with attributes attached to its nodes and edges) as a point in a suitable vector space; preserving the similarity of the graphs i.e. the more two graphs are considered similar, the closer should be the corresponding points in the vector space.

Fuzzy Graph Embedding (FGE): Fuzzy Graph Embedding (FGE) is a method of explicit graph embedding, which we recently proposed in [9]. FGE exploits structural and statistical significant details of an attributed graph for embedding it into a feature vector. Its resulting feature vector is termed as Fuzzy Structural Feature Vector (FSFV). FSFV contains graph level features, node level features and edge level features. The graph level features are graph order and graph size. Node level features are fuzzy histogram of node degree and fuzzy histogram of each distribution of the values taken by node attributes. And the edge level features are fuzzy histogram of each distribution of the values taken by edge attributes. FGE employs fuzzy overlapping trapezoidal intervals for minimizing the information loss while mapping from continuous graph space to discrete vector space. FGE permits graphs to benefit from state of the art computational efficient, clustering and classification tools, which originally is not possible in graph space.

III. Proposed method

A. Automatic indexation of a graph repository

The automatic indexation of a graph repository is performed during the off-line unsupervised learning phase of our proposed method. This phase is outlined in Figure 1 and is detailed in succeeding paragraphs of this section. The input to the (off-line unsupervised) learning phase is a collection of graphs, given by:

$$\{AG_1, AG_2, ..., AG_k, ..., AG_n\}$$

where, the k^{th} graph is:

$$AG_k = (V_k, E_k, \mu^{V_k}, \mu^{E_k})$$

First of all, these graphs are preprocessed and new resemblance attributes are added to nodes and edges of the graphs. These resemblance attributes incorporate information on the homogeneity in neighborhood of nodes and edges.

For an edge, resemblance attributes provide a measure of homogeneity between its nodes, and are computed from respective node attributes in the graph. For an edge between nodes, "1" and "2", resemblance between a numeric attribute

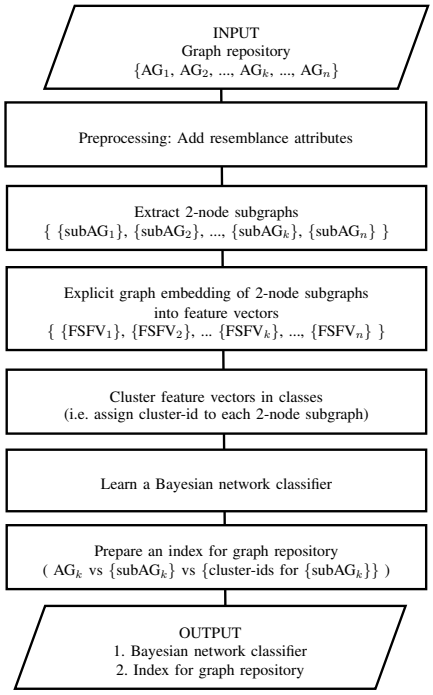


Figure 1. Automatic indexation of a graph repository.

"a" is computed by Equation 1 and resemblance between a symbolic attribute "b" is computed by Equation 2.

$$resemblance = min(|a_1|, |a_2|)/max(|a_1|, |a_2|) \quad (1)$$

$$resemblance = \begin{vmatrix} 1 & b_1 = b_2 \\ 0 & otherwise \end{vmatrix} \quad (2)$$

Similarly for a node, the resemblance attributes provide a measure of homogeneity among its edges, and are computed from the respective edge attributes in the graph. For a node, a resemblance attribute is computed as the mean of the resemblance between all pairs of its edges. The latter is computed by Equation 1 for a numeric attribute "a" and by Equation 2 for a symbolic attribute "b".

The next step in (off-line unsupervised) learning phase of our method, extracts 2-node subgraphs for all graphs in the input collection of graphs. The set of 2-node subgraphs for the input collection of graphs is given by:

$$\{\{subAG_1\}, \{subAG_2\}, ..., \{subAG_k\}, ..., \{subAG_n\}\}$$

where the set of subgraphs for k^{th} input graph (AG_k) is:

$$subAG_k = \{subAG_k^1, subAG_k^2, ..., subAG_k^i\}$$

and i^{th} 2-node subgraph for k^{th} input graph (AG_k) is:

$$subAG_k^i = (V_k^i, E_k^i, \mu^{V_k^i}, \mu^{E_k^i})$$

The next step in (off-line unsupervised) learning phase of our method, embeds the set of 2-node subgraphs by equal size feature vectors (the FSFVs). This is achieved by our explicit graph embedding method (the FGE). The set of FSFVs for the input collection of graphs is given by:

$$\{\{FSFV_1\}, \{FSFV_2\}, ..., \{FSFV_k\}, ..., \{FSFV_n\}\}$$

where, set of feature vectors for the 2-node subgraphs of k^{th} input graph (AG_k) is:

$$FSFV_k = \{FSFV_k^1, FSFV_k^2, ..., FSFV_k^i\}$$

and the i^{th} 2-node subgraph for k^{th} input graph (AG_k) is embeded by feature vector $FSFV_k^i$.

The feature vectors are clustered into classes by an agglomerative (also called hierarchical) clustering method. The clustering process starts by computing a pairwise city-block-distance metric for the features in FSFV and builds a linkage tree using the single link algorithm. We use a method from [10] for getting an optimal cutoff for clusters - [10] is based on an econometric approach to verify that variables in multiple regression are linearly independent. The use of agglomerative clustering approach keeps our method free of any parameter for number of clusters. Each cluster represents (a class of) similar 2-node subgraphs. The clustering step of (off-line unsupervised) learning phase assigns class labels to the FSFVs of 2-node subgraphs.

The labeled FSFVs are employed for learning a Bayesian network classifier. This classifier serves as a computational efficient tool for recognizing the 2-node subgraphs in query subgraph, during the subgraph spotting phase of our method. And finally, as a last step of (off-line unsupervised) learning phase of our method, an index is constructed for the input collection of graphs which maps a graph to 2-node subgraphs and the latter to cluster labels.

B. Subgraph spotting

The subgraph spotting is performed during on-line querying phase. The input to (on-line) querying phase is a graph comprising of at-least one 2-node subgraph. The input graph (AG_q) passes through the steps of preprocessing, extraction of 2-node subgraphs ($\{subAG_q\}$) and explicit graph embedding of 2-node subgraphs into feature vectors ($\{FSFV_q\}$), as detailed in previous section for automatic indexation of graph repository.

As next step of (on-line) querying phase, each query 2-node subgraph ($\{subAG_q\}$) is classified (as one of the 2-node subgraph clusters) by employing the Bayesian network classifier. For each query 2-node subgraph, the Bayesian network classifier outputs a list of posterior-probabilities for all clusters. The query 2-node subgraph is classified as highest posterior-probability cluster. We look-up this cluster-id in repository index for getting a list of possible combinations of 2-node subgraphs corresponding to query. The graphs containing all the query 2-node subgraphs ($\{subAG_q\}$) form the set of result graphs for the query graph AG_q. This result graph set is given by:

$$\{AG_1, AG_2, ..., AG_k, ..., AG_m\}$$

where, the k^{th} result graph is:

$$AG_k = (V_k, E_k, \mu^{V_k}, \mu^{E_k})$$

For spotting the query graph AG_q in a result graph AG_k, we employ the adjacency matrix of graph AG_k along-with a score function. For two nodes "i" and "j", the possible values in the adjacency matrix are summarized in Equation 3. The adjacency matrix of graph AG_k has a value of "0" if there is no edge between "i" and "j" in the original graph AG_k, a value of "1" if there is an edge between "i" and "j" in the original graph AG_k and a value of "2" if one of the query 2-node subgraphs is classified (by Bayesian network) as belonging to the cluster of this 2-node subgraph (which is comprising of edge between "i" and "j").

$$AG_k(i,j) = \begin{vmatrix} 0 & \text{no edge between i and j} \\ 1 & \text{an edge between i and j} \\ 2 & \text{2-node subgraph in result} \end{vmatrix} \quad (3)$$

The query graph AG_q is finally spotted in the result graph AG_k, by looking up in the neighborhood of each 2-node subgraph of AG_k which is in the result i.e. each "$AG_k(i,j) = 2$" in the adjacency matrix of result graph AG_k. We explore "w" connected neighbors of each "$AG_k(i,j) = 2$". The parameter "w" is proportional to the graph order of query graph AG_q ($|V_q|$). We compute a score for each "$AG_k(i,j) = 2$" using Equation 4. The computed score of "$AG_k(i,j) = 2$" also gives a confidence value for subgraph spotting of query graph AG_q in result graph AG_k.

$$score = \sum_{z=0}^{2} (z \times \frac{|z|}{w}) \quad (4)$$

In Equation 4,

z is a value in the adjacency matrix (either 0,1 or 2),

$|z|$ is number of times the value z occurs in neighborhood and

w is number of connected neighbors that are looked-up.

IV. Experimentation

We have evaluated the performance measures for our proposed method on two graphic document image repositories. The line-drawing graphic document images and the queries are generated synthetically along with the ground truth [11]. The document images are represented by attributed graphs using the method from Qureshi et al. [12]. The topological and geometric details about the structure of the line-drawing graphic content are extracted and represented by an attributed graph. In first step, the graphic content is vectorized and is represented by a set of primitives (vectors and quadrilateral). In next step, these primitives are represented by nodes and topological relations between them by edges in an attributed graph.

The synthetically generated query images for our experimentation, actually simulate the contextual noise. This type of noise occurs in cropped regions of graphic document images. An interesting application of the latter is selecting a region in a graphic document image (on a modern tactile interface), for querying a graphic document CBIR system.

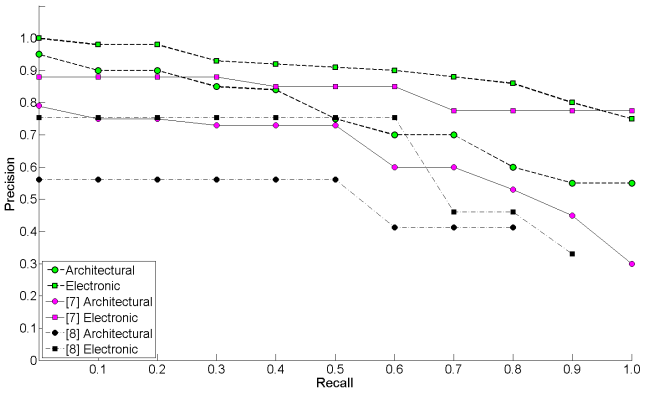


Figure 2. Precision and recall plot.

Table I
Dataset details.

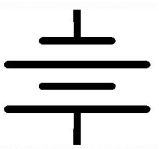
	Image		Attributed graph	
Electronic diagrams	Backgrounds	8	Avg. order	212
	Models	21	Avg. size	363
	Symbols	9600	Node attribs.	4
			Edge attribs.	3
	Documents	800	Graphs	800
	Queries	1000	Graphs	1000
Architectural diagrams	Backgrounds	2	Avg. order	359
	Models	16	Avg. size	733
	Symbols	4216	Node attribs.	4
			Edge attribs.	3
	Documents	200	Graphs	200
	Queries	1000	Graphs	1000

The details on the document images and graph datasets are summarized in Table I. During the automatic indexation phase of graph repositories, a total of 516714 2-node subgraphs were extracted for electronic diagrams and 305824 2-node subgraphs for architectural diagrams. These 2-node subgraphs were clustered into 455 classes for electronic diagrams and 211 classes for architectural diagrams. The indexation of electronic diagrams graph repository took ~17 hours on a standard PC with 2GB of RAM and the subgraph spotting was performed in real-time.

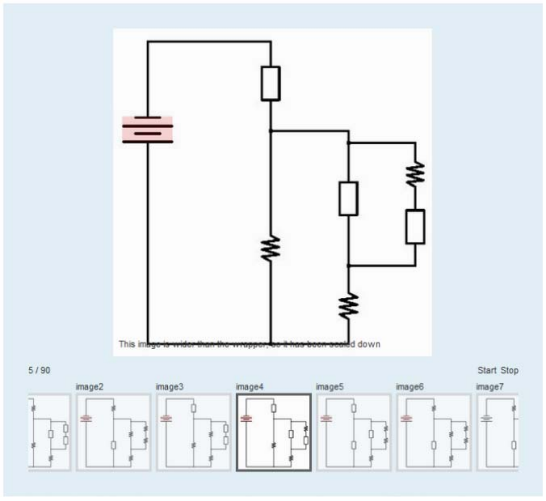
For evaluating the performance of our method, we have employed the standard precision and recall measure. Figure 2 presents the precision and recall plot for the central tendency of the queries of respective attributed graph datasets. A snapshot of the retrieved results for a query image is presented in Figure 3. The experimental results clearly demonstrate that the proposed method is capable of maintaining a high precision rate for sufficiently large graph repositories.

V. Conclusion

We have presented a method of spotting a subgraph in a graph repository. Subgraph spotting is a very interesting research problem for various application domains. This is a



(a) Query image.



(b) Documents retrieved.

Figure 3. A snapshot of retrieved results for a query image.

continuation of our work on content spotting in graph representation of line-drawing graphic document images. Our proposed method accomplishes subgraph spotting through graph embedding. We achieve automatic indexation of a graph repository during off-line learning phase and achieve the subgraph spotting during on-line querying phase.

Our proposed method does not rely on any domain specific details and offers a very general solution to the problem of subgraph spotting; indeed equally applicable to a wide range of application domains where the use of graph as a data structure is mandatory. Apart from incorporating learning abilities in structural representations (without requiring any labeled training set) and offering the ease of query by example (QBE) and the granularity of focused retrieval, the system does not impose any strict restrictions on the size of query subgraph. Our novel proposed system offers an improvement to the indexation part of off-line unsupervised learning phase of our previous work and gives a very general solution for subgraph spotting. The fact that our system does not require any labeled training-set enables its less expensive and fast deployment to a wide range of application domains.

The experimental results for graphic document image repositories are very encouraging and offer an improvement to our previous system. In future we plan to evaluate the performance of our method on real datasets for important application domains of pattern recognition and to take this work forward by exploring cliques of higher order (≥ 3) for building a multi-resolution index of a graph repository.

ACKNOWLEDGMENT

This work was partially supported by the Spanish projects TIN2008-04998, TIN2009-14633-C03-03 & CSD2007-00018 and partially by the PhD grant PD-2007-1/Overseas/FR/HEC/222 from Higher Education Commission of Pakistan.

REFERENCES

[1] J. Lladós, E. Valveny, G. Sánchez, and E. Martí, "Symbol Recognition: Current Advances and Perspectives," in *Graphics Recognition Algorithms and Applications*, 2002, vol. 2390, pp. 104–128.

[2] K. Tombre, S. Tabbone, and P. Dosch, "Musings on symbol recognition," in *Graphics Recognition. Ten Years Review and Future Perspectives*, 2006, pp. 23–34.

[3] D. Conte, P. Foggia, C. Sansone, and M. Vento, "Thirty years of graph matching in pattern recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 18, no. 3, pp. 265–298, 2004.

[4] H. Bunke and K. Riesen, "Recent advances in graph-based pattern recognition with applications in document analysis," *Pattern Recognition*, vol. 44, no. 5, pp. 1057–1067, May 2011.

[5] M. Rusinol and J. Lladós, *Symbol Spotting in Digital Libraries*. Springer, 2010.

[6] R. Raveaux, J.-C. Burie, and J.-M. Ogier, "A graph matching method and a graph matching distance based on subgraph assignments," *Pattern Recognition Letters*, vol. In Press,, no. 5, pp. 394–406, 2009.

[7] M. M. Luqman, T. Brouard, J.-y. Ramel, and J. Llados, "A Content Spotting System For Line Drawing Graphic Document Images," in *2010 International Conference on Pattern Recognition*, vol. 20, 2010, pp. 3420–3423.

[8] R. J. Qureshi, J.-Y. Ramel, D. Barret, and H. Cardot, "Spotting Symbols in Line Drawing Images Using Graph Representations," in *Graphics RECognition*, 2007, pp. 91–103.

[9] M. Luqman, J. Lladós, J.-Y. Ramel, and T. Brouard, "A Fuzzy-Interval Based Approach for Explicit Graph Embedding," in *Recognizing Patterns in Signals, Speech, Images and Videos*, 2010, vol. 6388, pp. 93–98.

[10] Y. Okada, T. Sahara, S. Ohgiya, and T. Nagashima, "Detection of cluster boundary in microarray data by reference to mips functional catalogue database," in *International Conference on Genome Informatics*, 2005, pp. 2–3.

[11] M. Delalandre, E. Valveny, T. Pridmore, and D. Karatzas, "Generation of synthetic documents for performance evaluation of symbol recognition & spotting systems," *International journal on document analysis and recognition*, pp. 1–21, 2010.

[12] R. J. Qureshi, J.-Y. Ramel, H. Cardot, and P. Mukherji, "Combination of Symbolic and Statistical Features for Symbols Recognition," in *International Conference on Signal Processing, Communications and Networking*, Feb. 2007, pp. 477–482.