On the Computation of the Common Labelling of a Set of Attributed Graphs

Albert Solé-Ribalta and Francesc Serratosa

Department of Computer Science and Mathematics
Universitat Rovira i Virgili (URV). Avda. Països Catalans, 26. 43007 Tarragona
albert.sole@urv.cat, francesc.serratosa@urv.cat

Abstract. In some methodologies, it is needed a consistent common labelling between the vertices of a set of graphs, for instance, to compute a representative of a set of graphs. This is a NP-problem with an exponential computational cost depending on the number of nodes and the number of graphs. The aim of this paper is twofold. On one hand, we aim to establish a technical methodology to define this problem for the present and further research. On the other hand, we present two sub-optimal algorithms to compute the labelling between a set of graphs. Results show that our new algorithms are able to find a consistent common labelling while reducing, most of the times, the mean distance of the AG set.

Keywords: Multiple graph matching, common graph labelling, inconsistent labelling, softassign.

1 Introduction

In some patter recognition applications, it is useful to define a representative of a set or cluster of elements. Some well-known techniques have been described when the elements are characterised by a feature vector. Nevertheless, only few techniques have been developed when the elements are represented by Attributed Graphs (AGs). When this is the case and when we want to synthesise the representative, examples could be found in: [1], [2], [3], [4] and [7], considering all the set at a time, it is needed a common labelling between each AG vertex (and arcs) and the vertices (and arcs) of the representative. Thus, given a priori this labelling, the new representative can be synthesised. Moreover, if we want the new structure to represent the cluster, it is desired that this structure is defined such that the sum of distances between the AGs and this new prototype is minimum. When this occurs, we say that we have obtained an Optimal Common Labelling of a set of AGs (OCL).

The main impediment on solving the OCL problem is that it is an NP-problem and therefore, the computational cost is exponential on the number of nodes and also on the number of AGs. For this reason, it is crucial to find algorithms that compute good approximations of the OCL in polynomial time. The aim of this paper is to present two new sub-optimal algorithms to compute the OCL of a set of AGs.

Similar works on this issue could be found in [8] where a labeling between two AG is induced by all the labelings of the set, however no OCL can be found using this

E. Bayro-Corrochano and J.-O. Eklundh (Eds.): CIARP 2009, LNCS 5856, pp. 137–144, 2009.

procedure. Some other works, where the aim of the article is finding the OCL, will we be introduced later on the document.

The document is structured as follows. In the next section, we introduce basic concepts and notation related to AG matching. In section 3, we define and explain, in detail, the problem we want to solve. In section 4, the new algorithms are briefly introduced. Section 5 presents results and evaluation of the algorithms. Finally, section 6 summarizes the article with some conclusions and further work.

2 Definitions

Definition 1. Attributed Graph: Let Δ_v and Δ_e denote the domains of possible values for attributed vertices and arcs, respectively. An attributed graph AG over $(\Delta_v$ *and* $\Delta_e)$ is defined by a tuple $AG=(\Sigma_v, \Sigma_e, \gamma_v, \gamma_e)$, where $\Sigma_v= \{v_k \mid k = 1,\ldots,R\}$ is the set of vertices (or nodes), $\Sigma_e= \{e_{ij} \mid i,j \in \{1,\ldots,R\}, i \neq j\}$ is the set of arcs (or edges) and $\gamma_v: \Sigma_v \rightarrow \Delta_v$, $\gamma_e : \Sigma_e \rightarrow \Delta_e$ assign attribute values to vertices and arcs respectively.

Definition 2. Isomorphism between AGs: Let $G^p = (\Sigma_v^p, \Sigma_e^p, \gamma_v^p, \gamma_e^p)$ and $G^q=(\Sigma_v^q, \Sigma_e^q, \gamma_v^q, \gamma_e^q)$ be two AGs. Moreover, let T be a set of isomorphisms between two vertex sets Σ_v. The isomorphism $f^{pq}: \Sigma_v^p \rightarrow \Sigma_v^q$, $f^{pq} \in \mathrm{T}$, assigns each vertex from G^p to only one vertex of G^q. There is no need to define the arcs isomorphism since they are mapped accordingly to the node isomorphism of their terminal nodes.

Definition 3. Cost and Distance between AGs: Let f^{pq} be the isomorphism $f^{pq}: \Sigma_v^p \rightarrow \Sigma_v^q$ that assigns each vertex from G^p to a vertex of G^q. The cost of this isomorphism, $C(G^p, G^q, f^{pq})$ is a function that represents how similar are the AGs and how correct is the isomorphism. Usually, $C=0$ represents that both AGs are identical and that the isomorphism captures this similarity. The distance D between two AGs, is defined to be the minimum cost of all possible isomorphisms f^{pq}. That is, $D\left(G^p, G^q\right) = \min_{f^{pq} \in \mathrm{Y}} C\left(G^p, G^q, f^{pq}\right)$ [9]. We say that the isomorphism f^{pq} is *optimal* if it is the one used to compute the distance.

3 Common Labelling of a Set of AGs

The first step of the algorithms presented in the literature [1], [2], [3], [4] is to obtain all possible isomorphisms between all AGs of the set. Once these isomorphisms are obtained, then the Common Labelling is computed.

Definition 4. Multiple Isomorphism of a set of AGs: Let S be a set of N AGs, $S=\{G^1, G^2, \ldots, G^N\}$. We say that the set F is a Multiple Isomorphism of S if F contains one and only one isomorphism between the AGs in S, $F = \{f^{1,2}, \ldots, f^{2,1}, \ldots, f^{N,N}\}$.

We assume that the AGs have R nodes. If it is not the case, the AGs would have to be extended with null nodes. We say that a multiple isomorphism is *consistent* if concatenating all the isomorphisms, we can define disjoint *partitions* of vertices. Every *partition* is supposed to contain one and only one vertex per each AG and, in

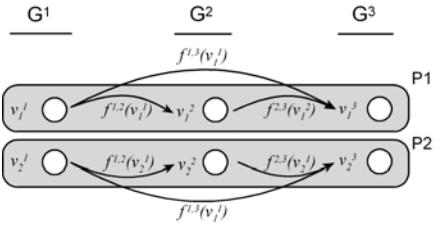

Fig. 1. Consistent multiple isomorphism

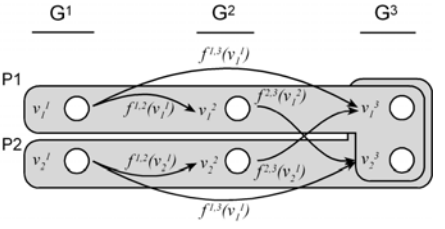

Fig. 2. Inconsistent multiple isomorphism

addition, every vertex must belong to only one partition. **Figure 1** shows a *Consistent Multiple Isomorphism* between three AGs, being $R=2$. We can distinguish two partitions, *P1* and *P2*. **Figure 2** shows the same AGs with an *Inconsistent Multiple Isomorphism*, where partitions share two nodes, hence partitions are not disjoint.

Definition 5. Consistent Multiple Isomorphism of a set of AGs (CMI): Let F be a Multiple Isomorphism of S. F is a CMI of S if it fulfils that $f^{qk}\left(f^{pq}\left(v_i^{pq}\right)\right)= f^{pk}\left(v_i^{pk}\right)$, $0 < p,q,k \le N, 0 < i \le R$.

Given an isomorphism, we can define its costs (**definition 3**). Extending this definition, given a CMI of a set, we define its cost as the addition of the costs of all isomorphisms. The Optimal Consistent Multiple Isomorphism (OCMI) is the CMI with the minimum cost. Note that, the cost of the OCMI may be obtained by non-optimal isomorphisms since it is restricted to be consistent.

Definition 6. Optimal Consistent Multiple Isomorphism of a set of AGs (OCMI): Let F be a CMI of S. F is an Optimal Consistent Multiple Isomorphism (OCMI) of S if it fulfils that $F = \arg\min_{f^{pq}\in \Upsilon} \sum_{\forall G^p, G^q} C\left(G^p, G^q, f^{pq}\right)$.

Given a CMI, we can define a Common Labelling (CL) of a set of AGs. Note that all the vertices of each partition are labelled to the same node of the virtual structure. For this reason, it is needed the MI to be consistent. If not, the CL would not be a function since an AG node would have to be labelled to several nodes of the virtual structure.

Definition 7. Common Labelling of a set of AGs (CL): Let F be a CMI of S and let L_v be a vertex set, $L_v \in \Sigma_v$. The Common Labelling $H= \{ h^1, h^2, \ldots , h^n\}$ is defined to be a set of bijective mappings from the vertices of AGs to L_v as follows: $h^1(v_i^1)=i$ and $h^p(v_i^p)=h^{p-1}(v_j^{p-1})$, $1 \le i,j \le R$, $2 \le p \le N$, being $f^{p-1,p}(v_j^{p-1})=v_i^p$. **Figure 3** illustrates this definition.

Finally, the Optimal Common Labelling of a set is a CL computed through an OCMI. The prototype or representative of the set synthesised using this CL would be the best representative, from the statistical point of view, since the sum of the costs of each pair of AGs, considering the global consistency requirement, is the lowest among all possible CL.

Definition 8. Optimal Common Labelling of a set of AGs (OCL): Let H be a CL of S computed by a CMI F. We say that H is an Optimal Common Labelling (OCL) of S if F is an OCMI of S.

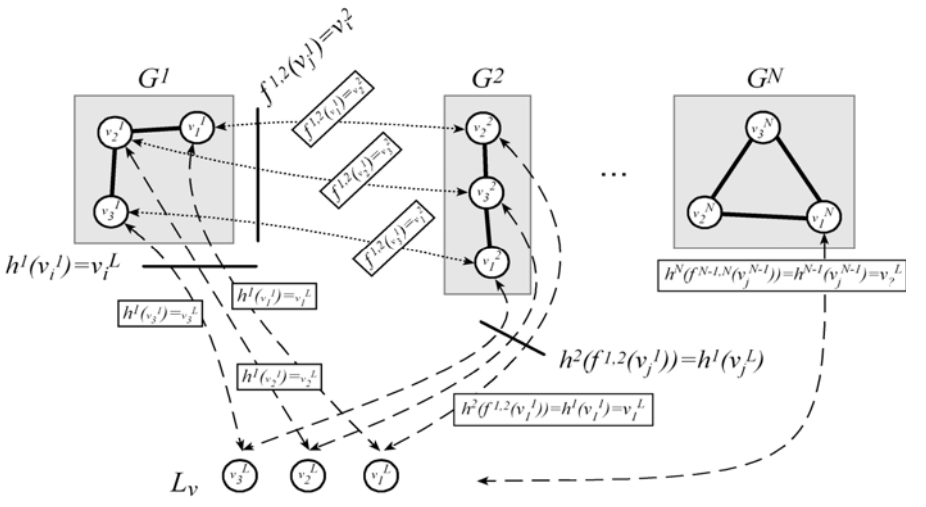

Fig. 3. Illustration of a CL given a CMI

Consistency Index

Through **definition 5**, we can discern whether a MI is consistent or not. Nevertheless, we are interested in establishing a consistency measure of a non-consistent MI to know the goodness of a concrete labelling given by sub-optimal labeling algorithms.

The *consistency index* that we propose shows the correctness of a MI taking values in the domain [0,1]. The higher values obtained the better consistency in the MI. Only in the case that the MI is consistent, the consistency index equals 1. To obtain a smooth index, we base our index on the number of inconsistencies given any triplet of AGs from the set. Thus, given the triplet of AGs, G^1, G^2 and G^3 and the MI $F = \{f^{1,2}, f^{1,3}, f^{2,3}\}$, we say that there is a tripled inconsistency if $f^{1,3}\left(v_i^1\right) \neq f^{2,3}\left(f^{1,2}\left(v_i^1\right)\right)$, $1 \leq i \leq R$ (extracted from **definition 5**). The cost of this operation is linear respect the number of nodes. Finally, the *Consistency Index* is obtained as a function depending on the number of triplet inconsistencies and the number of possible triplets.

$$Consistency\ Index = 1 - \left|Tripled\ Inconsistency\right| \Big/ \binom{N}{3}$$

4 Computing the OCL

Figure 4 presents the main methodology used up to now to compute a sub-optimal solution for the Common Labelling problem[1][2][3][4]. It is composed by three main steps. First, for each pair of AGs, an assignation matrix is computed. To do so, several error-tolerant graph matching algorithms have been presented, such as, probabilistic relaxation [11], softassign [5] or Expectation-Maximisation [12]. Each cell of the assignation matrix M_{ai} stores the probability of node *a,* from G^1, to be assigned to node *i*, from graph G^2, that is, the probability of the labelling $f^{1,2}\left(v_i^1\right) = v_a^2$, $p\left(v_i^1, v_a^2\right)$.

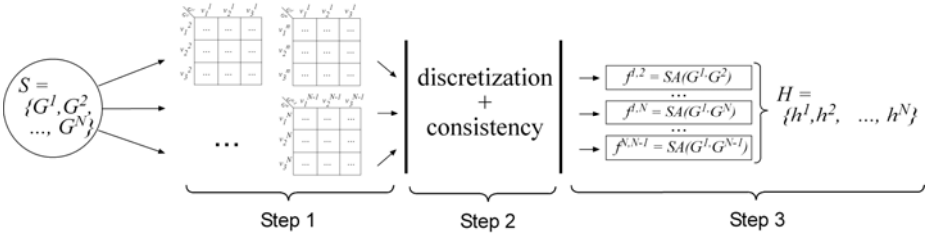

Fig. 4. Basic scheme to compute a CL based on a set of assignation matrices

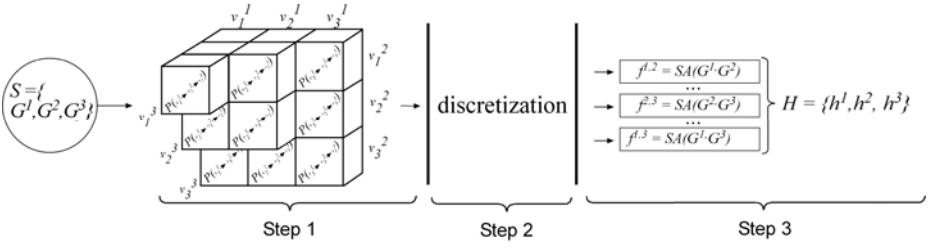

Fig. 5. New scheme to compute a CL based on an assignation hypercube. Example using N=3 graphs.

The cost to obtain this matrix, using softassign [5], is $O((N/2\text{-}N)\cdot R^4)$ per iteration of the algorithm.

In the second step, some times called *clean-up* process, the CMI of the set is obtained. Again, several techniques can be used, but, all of them consider the probabilities of all the individual assignation matrices and the restrictions imposed by the consistency requirements to obtain the final CMI. The cost of this step, again using softassign, is $O((N/2\text{-}N)^R)$. Finally, in the last and simplest step, the CL is defined.

Figure 5 presents our new methodology. The main difference appears in the first and second step. In the first step, the set of assignation matrices is substituted by an assignation hypercube to alleviate the problem of taking the individual assignation matrices independently. As a consequence, the first step generates always consistent isomorphisms. Therefore, the second step does not need to figure out a consistent MI. The third step is equivalent to the previous methodologies. The number of dimensions of the hypercube is the number of AGs of the set, *N*. Each cell of the hypercube $M_{a_1 a_2 \ldots a_N}$ represents the joint probability of:

$$f^{1,2}\left(v_{a_1}^1\right)=v_{a_2}^2 \;\&\; f^{1,3}\left(v_{a_1}^1\right)=v_{a_3}^3 \;\&\ldots\&\; f^{1,N}\left(v_{a_1}^1\right)=v_{a_N}^N \;\&\ldots\&\; f^{2,3}\left(v_{a_2}^2\right)=v_{a_3}^3 \;\&\ldots\&\; f^{2,N}\left(v_{a_2}^2\right)=v_{a_N}^N \ldots$$

That is, $p\left(v_{a_1}^1, v_{a_2}^2, \ldots, v_{a_N}^N\right)$.

Step 1 of the new algorithm: computing an assignation hyper-cube

To compute the assignation hyper-cube, we have developed two algorithms. In the first one, called *N-dimensional softassign*, the joint probability $p\left(v_{a_1}^1, v_{a_2}^2, \ldots, v_{a_N}^N\right)$ has to be computed all at once since the marginal probabilities are not considered independent. The algorithm is a generalisation of the softassign algorithm, in which, the

double-stochastic [6] matrix is converted to an N-stochastic hyper-cube and all other parts of the algorithm are extended accordingly. It has the advantage that the whole MI is considered at once in each iteration; and therefore, the information of the partial labelings is used globally. The computational cost is $O(R^{2N})$ at each iteration.

The second algorithm, called *agglomerative softassign*, is based on the supposition that the joint probability $p\left(v_{a_1}^1, v_{a_2}^2, ..., v_{a_N}^N\right)$ can be obtained as the product of the marginal ones since they are independent:

$$p\left(v_{a_1}^1, v_{a_2}^2, ..., v_{a_N}^N\right) = p\left(v_{a_1}^1, v_{a_2}^2\right) \times p\left(v_{a_1}^1, v_{a_3}^3\right) .. p\left(v_{a_1}^1, v_{a_N}^N\right) \times p\left(v_{a_2}^2, v_{a_3}^3\right) .. p\left(v_{a_2}^2, v_{a_N}^N\right) ..$$

The algorithm is composed by two main steps. In the first one, all the individual assignation matrices are computed using any of the algorithms mentioned before. Using those individual assignation matrices, in the second step, the cells of the hypercube are obtained as the product of the corresponding cells of the assignation matrices.

$$M_{v_{a1}^{G1}, v_{a2}^{G2}, ..., v_{aN}^{GN}} = \prod_{1 \le i < j \le N} M_{a_i, a_j}^{Gi, Gj}$$

The cost is the sum of computing all individual assignation matrices plus the cost of joining all those individual matrices to the hypercube $O(((N^2/2)-N) \cdot R^N)$.

In this paper we base our extensions of "Step 1" on the softassign algorithm due to the *N-dimensional* procedure must extend a concrete algorithm to compute the joint probability of several graphs. However, both procedures are generic enough to be applied to any other algorithms that use a probabilistic approach, e.g. [8] and [11].

5 Evaluation

We have evaluated the model using a dataset created at the University of Bern [10]. It is composed of 15 capital letters (classes) of the Roman alphabet i.e. A, E, F, H, I, K, L, M, N, T, V, W, X, Y and Z. Each letter is constituted by straight lines which represent edges and terminal points which represent nodes. Nodes are defined over a two-dimensional domain that represents the position (x, y) in the plane. Edges have a one-dimensional and binary attribute that represents the existence or non-existence of a line between two terminal points. Graph-based representations of the prototypes are shown in **Figure 6**. This database contains three sub-databases with different distortion level: low, med, high. Each distortion level of each letter is composed by 150 examples. **Figure 7** shows 3 examples of letter X with low distortion and 3 examples of letter H with high distortion. In this evaluation just high distortion has been used.

To evaluate the new methodologies proposed we have performed two experiments. The aim of the first experiment is to evaluate the consistency of the MI obtained in

Fig. 6. Graph-based representations of the original prototypes

Fig. 7. X and H examples with with low and high distortion level respectively

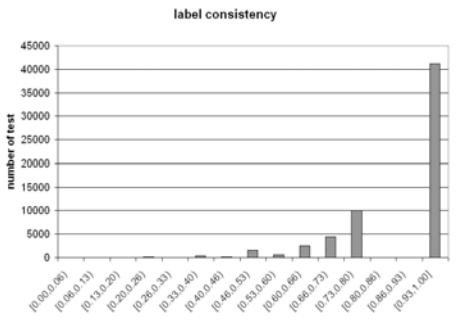

Fig. 8. Label consistency of labelings found using classical softassign

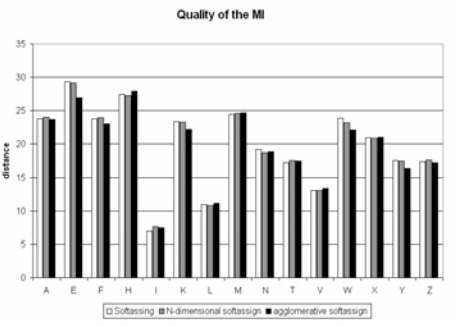

Fig. 9. Quality of the MI using the two algorithms presented and the softassign

step 1 of the basic scheme[1] (Figure 4). The second experiment is addressed to evaluate the goodness of CMI generated by the two schemes (Figure 4 and Figure 5)[2].

To perform the experiment we compute the CMI of a set S composed by three elements. We took 30 AGs of each database class. Using each set, we evaluated all possible labelings resulting 30*29 labelings. With all those labelings, we chose all possible triplets without repeating twice the same graph[3], generating ≈ 4000 triplets. We evaluated the *labeling consistency* for each triplet. The overall results for all test elements are shown in **Figure 8**. It can be observed that approximately 70% of the triplets don't produce labeling errors (the labeling consistency is aprox. 1). The other 30% of the triplets have different levels of consistency.

In the second experiment we compare the mean edit distance [9] of all test elements for each class. We took as a test set the inconsistent triplets found in experiment one. Due to temporal requirement of the *N-dimensional softassign* algorithm, we choose, randomly, a sub-set of 50 elements for each class.

Figure 9 shows the results for the second experiment. We observe that the mean of edit distances is approximately the same for the three methodologies. Therefore, we can conclude that it is possible to eliminate the inconsistencies obtained with the basic scheme without reducing the quality of the MI.

To summarize, it is worth to say that the expected results should reflect that when the basic scheme finds inconsistencies in the labelings, the proposed algorithms reduce to none those inconsistencies at the cost of augmenting the mean distance of the set S. However, we can amazingly see that in most of those cases where the basic scheme finds inconsistencies, the new scheme obtains better MI than the basic scheme while reducing to none the inconsistencies of the MI.

---

[1] Using the softassign algorithm.

[2] Using the softassign algorithm on the first scheme and the two presented methods in the second scheme.

[3] That is, only result where $(x, y, z) \mid (x \neq y) \wedge (x \neq z) \wedge (y \neq z)$ are produced.

6 Conclusions and Further Work

In this paper, we have presented two new approaches to compute a common labelling between a set of AGs. The main idea of these algorithms is to tackle the problem from the joint-probability point of view. The joint probability represents the probability of all the labelings between all the AGs taken at once. Up to now, the probabilities of each labelling had been considered as independent items and only at the end of the process, when the consistency had to be fulfilled, there where taken all together. Results show that it is possible to remove inconsistencies in marginal labeling function without incrementing the mean distance of the AG set.

Acknowledgements

This research supported by Consolider Ingenio 2010; project CSD2007-00018, by the CICYT project DPI 2007-61452 and by the Universitat Rovira I Virgili (URV) through a PhD research grant.

References

[1] Bonev, B., et al.: Constellations and the unsupervised learning of graphs. In: Escolano, F., Vento, M. (eds.) GbRPR. LNCS, vol. 4538, pp. 340–350. Springer, Heidelberg (2007)
[2] Sanfeliu, A., Serratosa, F., Alquézar, R.: Second-Order Random Graphs for modeling sets of Attributed Graphs and their application to object learning and recognition. International Journal of Pattern Recognition and Artificial Intelligence, IJPRAI 18(3), 375–396 (2004)
[3] Serratosa, F., Alquézar, R., Sanfeliu, A.: Function-Described Graphs for modeling objects represented by attributed graphs. Pattern Recognition 36(3), 781–798 (2003)
[4] Wong, A.K.C., You, M.: Entropy and distance of random graphs with application to structural pattern recognition. IEEE Transactions on PAMI 7, 599–609 (1985)
[5] Gold, S., Rangarajan, A.: A Graduated Assignment Algorithm for Graph Matching. Trans. on PAMI 18(4), 377–388 (1996)
[6] Latouche, G., Ramaswami, V.: Introduction to Matrix Analytic Methods in Stochastic Modelling. In: PH Distributions; ASA, ch. 2, 1st edn., SIAM, Philadelphia (1999)
[7] Solé-Ribalta, A., Serratosa, F.: A structural and semantic probabilistic model for matching and representing a set of graphs, GbR, Venice, Italy. LNCS, vol. 5534, pp. 164–173. Springer, Heidelberg (2009)
[8] Williams, M.L., Wilson, R.C., Hancock, E.R.: Multiple Graph Matching with Bayesian Inference. Pattern Recognition Letters 18, 1275–1281 (1997)
[9] Sanfeliu, A., Fu, K.S.: A Distance Measure Between Attributed Relational Graphs for Pattern Recognition. Trans. Systems, Man, and Cybernetics 13, 353–362 (1983)
[10] Riesen, K., Bunke, H.: Graph Database Repository for Graph Based Pattern Recognition and Machine Learning. In: SSPR 2008 (2008)
[11] Fekete, G., Eklundh, J.O., Rosenfeld, A.: Relaxation: Evaluation and Applications. Trans. on PAMI 4(3), 459–469 (1981)
[12] Luo, B., Hancock, E.R.: Structural graph matching using the EM algorithm and singular value decomposition. Trans. on PAMI 10(23), 1120–1136 (2001)