A Comparison of Least Squares and Spectral Methods for Attributed Graph Matching

Jianfeng Lu^{1,2}, Terry Caelli¹, and Jingyu Yang²

¹ Department of Computing Science, University of Alberta, Edmonton

Alberta, Canada, T6G 2E8

lujf@mail.njust.edu.cn, tcaelli@ualberta.ca

² Department of Computer Science, Nanjing University of Science & Technology
Nanjing, P.R. China, 210094

Abstract. In this paper, least squares and spectral methods for attributed graph matching are compared. For the least squares method, complete graphs and decomposed graph models are considered in conjunction with the least squares approximations to optimal permutation matrices. We have used a version of Umeyama's spectral method for comparison purposes. Results clearly demonstrate how both these methods are affected by additive noise but that, in general, least squares methods are superior.

1 Introduction

Graphs are a very powerful tool for many practical problems including pattern recognition and computer vision [1-8]. Graph matching algorithms can be divided into search-based methods and methods based on optimization [3]. Least squares (LSM) and spectral methods (SM) are two recent optimization methods for attributed graph matching. However, to this date, there has not been a direct comparison of their relative benefits and deficits.

The method proposed by van Wyk et al.[1] is a more recent formulation of the least squares method. The essence of their method is to construct an interpolation function to approximate the best fitting permutation matrix (Interpolator-Based Kronecker Product Graph Matching (IBKPGM)) that maps vertices of one graph into the other. However, the method is less than ideal under additive noise conditions. Consequently it makes sense to integrate this method with graph models that are known to scale well with noise, in particular, those based on decomposition models. El-Sonbaty and Ismail [2] proposed a decomposition-based method where isomorphisms are established between sets of subgraphs – to be explored within the context of the LSM in this paper. Grewe and Kak[11] also adopted the similar model to extract local feature set for 3D object recognition.

An alternative to least squares methods is the spectral methods that have also received recent attention [4-8]. The main advantage of SMs is their low complexity, but their disadvantage is, again, their sensitivity to noise. Umeyama [4] proposed an eigenvalue decomposition (EVD) approach for weighted graph matching. For two weighted graphs, eigenvalue decompositions were computed after which the product of the two eigenvector matrices were used as inputs to a linear assignment algorithm

to find the optimal isomorphism. Scott et al.[5] adopted a method that maximized the inner product of two affinity matrices to find the correspondence between two images. This approach was extended by Shapiro et al.[6]. For two image feature sets, the singular value decomposition (SVD) of their feature distance matrix was computed and the distances between each pair of eigenvectors were used to determine vertex correspondences. Kosinov and Caelli[8] proposed an eigen-subspace projection clustering method for inexact graph matching. The main idea is that for a pair of graphs (having the same or different number of vertices) their eigenvalue/vector decompositions are obtained and the vertices are then projected into a normalized eigenspectral subspace that adjusts for differences in vertex numbers and spectral energies. They then perform clustering in this normalized subspace to determine correspondences. However, extensions of SMs to attributed graph isomorphisms have not been reported in the literature.

In this paper, these two algebraic methods (SM, LSM) for attributed graph matching are compared. In addition, we propose an extension to the IBKPGM method based on using a piece-wise graph decomposition approach (the GDKPGM method). The layout of this paper is as follows. In Section 2, some preliminaries of the IBKPGM method are introduced. Extensions to current algorithms are proposed in Section 3. Some issues related to the attributed graph spectral method are discussed in Section 4, experimental results and comparisons are presented in Section 5. Section 6 draws conclusions about such algebraic methods.

2 The IBKPGM Method

The focus of IBKPGM is a matching model for attributed graphs where an input graph, G :

$$G = (V, E, \{A_i\}_{i=1}^r, \{B_j\}_{j=1}^s) \quad (1)$$

is matched to a reference graph, G' , where

$$G' = (V', E', \{A'_i\}_{i=1}^r, \{B'_j\}_{j=1}^s) \quad (2)$$

for $A_i \in R^{n \times n}$, $B_j \in R^{n \times 1}$, $A'_i \in R^{n' \times n'}$ and $B'_j \in R^{n' \times 1}$, representing the edge attribute matrices and vertex attribute vectors, respectively. The reference and input graphs each have r edge attributes and s vertex attributes while the number of vertices in G' and G are $n' := |V'|$ and $n := |V|$, respectively (see [1] for details).

Graph matching is then defined in terms of determining the permutation matrix, P , which minimizes

$$\min_P \left(\sum_{i=1}^r \left\| A_i - P A'_i P^T \right\|^q + \sum_{j=1}^s \left\| B_j - P B'_j \right\|^q \right) \quad (3)$$

where $\|\cdot\|$ denotes the Frobenius matrix norm with typical values for q of 1 or 2 and where P^T corresponds to the transpose of P . Although some proposed algorithms attempt to solve Eqn. (3) directly, Van Wyk et al [1] used the Kronecker Product form of Eqn. (3) to allow for a method that approximates P using an Interpolator-Based Kronecker Product formulation.

In this method, each edge attribute matrix of size of $n^2 \times n$ is transformed into a $n^2 \times 1$ column vector and each vertex attribute vector is transformed into a diagonal matrix first, and then, also, a vector. Thus, Eqn. (3) is rewritten as

$$\min_{\Phi} \left(\sum_{i=1}^{r+s} \left\| \text{vec} A_i - \Phi \text{vec} A_i' \right\|^q \right) \quad (4)$$

where $\Phi = P \otimes P$ and $\otimes$ denotes the Kronecker product of matrices, $\text{vec} Z$ is an operator that transforms a matrix Z into a vector. If the least squares method is applied to Eqn. (4) directly, its complexity is high, requiring the calculation of the pseudo-inverse of a $n^2 \times (r+s)$ matrix. To overcome this, van Wyk et. al.[1] used the interpolator method to derive an approximate optimal solution to Eqn. (4).

That is, if Eqn. (4) is viewed as the mapping $A_k = F(A_k')$, where $k = 1, \dots, r+s$, then the mapping of vertex i in A' to A can be approximated using an interpolator-based method described by

$$\tilde{F}_i(\cdot) = \sum_{l=1}^{r+s} C_{li} K(\text{vec} A_l', \cdot) \quad (5)$$

where $K(\cdot, \cdot)$ is the reproducing kernel. A common choice is a linear kernel:

$$K(\text{vec} X, \text{vec} Y) = (\text{vec} X)^T \text{vec} Y \quad (6)$$

In this case the interpolator coefficients, C_{li} , can be solved first by letting

$$A = CG \quad (7)$$

where

$$C = (C_1, C_2, \dots, C_{r+s}) \quad (8)$$

for

$$C_l = (C_{li}) \quad (9)$$

$$A = (\text{vec} A_1, \text{vec} A_2, \dots, \text{vec} A_{r+s}) \quad (10)$$

$$G = (G_{lk}) \quad (11)$$

and

$$G_{lk} = K(\text{vec} A_l', \text{vec} A_k') \quad (12)$$

Based on the estimated interpolator coefficients, C_{li} , the elements of Φ can be calculated by (see [1] for more details)

$$\hat{\Phi} = (\hat{\Phi}_1^T, \dots, \hat{\Phi}_{n'}^T)^T \text{ by } \hat{\Phi}_i^T = \left(\sum_{l=1}^{r+s} C_{li} \text{vec} A_l' \right)^T, \text{ where, } i = 1, \dots, n^2 \quad (13)$$

Since, $\Phi = P \otimes P$, an approximation to the permutation matrix, P , namely, $\bar{P}$, can be derived from $\hat{\Phi}$:

$$\bar{P}_{ij} = \frac{\sum_{kl} \hat{\Phi}_{kl}}{n}; \text{ where } i = 1, \dots, n; j = 1, \dots, n'; \quad (14)$$

$$k = (i-1)n + 1, \dots, (i-1)n + n; l = (j-1)n' + 1, \dots, (j-1)n' + n'$$

van Wyk et. al. [1] have shown that if an optimal assignment method is adopted, $\|P - \bar{P}\|_F$ is minimum in the mean-square-error sense, where F is the Frobenius norm (F-norm). Consequently, the permutation matrix, P , can be derived from $\bar{P}$ based on an assignment algorithm. Albeit, this method still lacks robustness to noise. For this reason we explore how to integrate the core concept of IBKPGM but over subgraphs of the graphs using the following simple decomposition model.

3 Graph Decompositions and GDKPGM Matching Algorithm

Here we have used the decomposition model proposed by El-Sonbaty and Ismail [2] where, for each vertex, v , a subgraph is defined consisting of all vertices that are directly connected (a path length of 1) to v . Fig. 1 shows such a simple decomposition (trees rooted by each vertex, see [2] for details).

Fig. 1. Original graph (a) and all subgraphs (b)-(e) after decomposition.

Suppose that after decomposition the set of reference graph subgraphs consists of $\{SGR_j\}, j=1, \dots, n'$ where n' corresponds to the number of vertices in the reference graph and SGR_j to the subgraph corresponding to vertex j . Similarly, $\{SGI_i\}, i=1, \dots, n$, corresponds to the subgraphs in the comparison (input) graph, having n vertices.

3.1 Algorithm

1. Decompose both the input and reference graphs - as above.
2. For each pair of SGR_j and SGI_i , compute replicator coefficients, C_{ij} for $A_i = C_{ij}G_j$ where A_i represents the attribute matrix of subgraph SGI_i by Eqn. (8) of size, $n^2 * (r + s)$; G_j is derived from SGR_j by Eqn. (11).
3. Compute $\bar{P}$ from Eqns. (13) and (14) and use the matrix F-norm to obtain F_{ij} , $F_{ij} = \|A_i - PA'_j P^T\|_F$ - the “cost” of matching the two subgraphs(vertices).
4. Use an optimal assignment method to find optimal correspondence between subgraph SGR_j and SGI_i .

3.2 Complexity Analysis

Discussion is limited to undirected, fully connected graphs and, for analysis convenience, suppose the two graphs have the same number vertices, namely, $n = n'$. The complexity of the IBKPGM method is $O(n^4)$ being dominated by the extraction of the Kronecker match matrix when $n \gg (r + s)$ from Eqns. (5)-(14). If we directly use their method, the complexity is high, $O(n^6)$. However, the proposed algorithm can reduce complexity. Based on the decomposition model, the edge attribute matrix of each subgraph has the following form: all diagonal elements are zero, and there is only one non-zero column and row - namely, the total number of non-zero elements is $2n - 2$. If this matrix is transformed into a column vector, there are $2n - 2$ non-zero rows and the vertex attribute matrix has only n non-zero diagonal elements. Hence, for each subgraph all attribute matrices are transformed into vectors and rearranged into an $n^2 * (r + s)$ matrix by Eqn. (8) having, at most, $3n - 2$ non-zero rows, where $2n - 2$ rows are from the edge attribute matrix and n rows from vertex attribute matrix.

According to Eqn. (12), the size of G_j is $(r + s) * (r + s)$, and the matrix A_i in Eqn. (12) has, at most, $3n - 2$ non-zero rows; therefore C_{ij} has at most $3n - 2$ non-zero rows. For Eqn. (13), since C and A_i have, at most, $3n - 2$ non-zero columns and rows their product has, at most, $3n - 2$ non-zero elements, so the actual complexity is $O(n^2)$ for Eqn. (13). Thus, the total non-zero elements in Φ should be, at most, $n^2 * (3n - 2)$, and the complexity of Eqn. (14) should be $O(n^3)$.

van Wyk's method [1] uses the Kuhn-Munkres optimal assignment algorithm to determine the final optimal correspondence between vertices. It has a complexity of $O(n^3)$ [9]. However, Yamada [10] designed an optimal assignment algorithm whose complexity is $O(n^2)$ which can be used to replace the Kuhn-Munkres method. Using this, then, the complexity of each loop of the GDKPGM algorithm is $O(n^3)$. Thus, the total complexity of this method is $O(n^5)$. It is one order higher than IBKPGM. For El-Sonbaty and Ismail's method [2], its complexity is, best case, $O(n^3)$; worst case, $O(n^5)$.

4 Spectral Methods for Graph Matching (SMGM)

Spectral methods typically consist of the following simple steps:

1. For attributed graphs, compute their vertex-wise covariance matrix (column-wise: $A^T A$). For simple non-attributed and undirected graphs, this corresponds to their adjacency matrices.
2. For the above matrices, eigenvalue/vector decomposition is performed.
3. Normalize common subspace bases (including sign correction or sign-independent comparisons) are computed for projecting vertices from both graphs.

4. Correspondence between the two graphs can be derived by clustering vertices in normalized common subspace (see Kosinov and Caelli [8]) or by solving an assignment problem.

Umeyama's method [4] is a representative example of this latter approach for the restricted case of weighted graphs. Here we have generalized his method for fully attributed vertices and edges. This generalization results in the following algorithm.

1. For both input and reference graph, the vertex (n) and edge attribute values (r, s respectively) are defined as a single matrix where each vertex attribute vector is formatted as a diagonal submatrix, edge attributes as off-diagonal matrices, all being concatenated as a single $((r + s) \times n) \times n$ matrix, A .
2. The vertex-based covariance matrix, $A^T A$, is computed.
3. Eigenvalue/vector decomposition is performed on $A^T A$ and a subspace dimension selected in terms of the first w eigenvalues/vectors.
4. Correlations between the eigenvectors are computed as a measure of "distance" ("cost") of matching vertices in this subspace.

For eigenvectors of $V1$ and $V2$, which are from input graph and reference graph respectively, the product of $|V2| * |V1^{-1}|$ is calculated to get a cost matrix.

5. The best set of correspondences is selected by solving an optimal assignment problem based on the cost matrix at step 4.

We have used this method for comparison with the LSM approaches.

5 Experiments

5.1 Random Graphs

We have compared the results of the GDKPGM, IBKPGM and SMGM methods using full-connected random graphs as used by Van Wyk et al.[1]. The parameters n', n, r, s were fixed for each iteration. A reference graph was then randomly generated with all attribute values distributed between 0 and 1. An $n \times n'$ permutation matrix, P , was randomly generated to permute the row and columns. An independently generated noise matrix, having uniformly generated values within the $[-\varepsilon, \varepsilon; 0 < \varepsilon < 1]$ interval was then added to each vertex and edge attribute. The task was then to match the original graph with the graph corresponding to the permuted, perturbed vertices and corresponding edges. The estimated probability of a correct vertex-vertex correspondence was calculated as a function of ε for every 100 experiments.

To evaluate performance we focused on the following questions.

- 1) Does the decomposition method improve performance
- 2) Is LSM always superior to SM?
- 3) How do the parameters, $\{n', n, r, s, \varepsilon\}$, influence the performance of above methods?

The following experiments were conducted to explore these issues. First, the attribute parameters, (r, s) , were set to $(3, 3)$, and (n', n) to $(10, 10)$, $(30, 30)$, $(50, 50)$

and (100,100). Second, $(r,s) = (1,0)$, and (n',n) were set to (10,10), (30,30), (50,50): a simple weighted graph. Third, $(r,s) = (1,1)$, and (n',n) were set to (10,10), (30,30), (50,50). Fourth, $(r,s) = (2,0)$, and (n',n) were set to (10,10), (30,30), (50,50). Fig. 2 shows results for the first experiment, and Tables 1-3 for the remaining.

5.2 Results

It can be seen in Fig. 2 that performance of the GDKPGM method is much better than that of IBKPGM. However, when ϵ is larger than 0.6, performance of GDKPGM also significantly decreases along with IBKPGM. Also, this decrease in performance increases with n , the number of vertices in the reference graph although, in all cases, GDKPGM performance is superior to IBKPGM. In this case, however, SM is inferior to LSM.

Fig. 2. Performance comparison of 3 methods, $r=s=3$, $(n',n)=(10,10)$, (30,30), (50,50), (100,100). (r,s) correspond to numbers of edge and vertex attributes; (n',n) : to number of vertices in reference and input graphs. A, B, C represent the GDKPGM, IBKPGM and SMGM methods, respectively.

Table 1 shows that, for a weighted graph ($r=1,s=0$), when the noise level is not large ($\epsilon < 0.3$), the SM is superior to LSM. When $\epsilon = 0.1$ the performance of SM is much better than that of LSM. But as the noise increases performance of SM decreases significantly, and when $\epsilon > 0.4$ it is quite inferior to the LSM methods.

Table 2 compares performance to that shown in Table 1 with an additional vertex attribute and demonstrates how additional constraints improved the performance of the LSM method but not the SM method, except for the low noise condi-

tions: $\varepsilon = 0.1$. The same conclusion can be draw from results shown in Table 3 when compared to Table 1. The difference in the experimental conditions for Table 2 and 3, (both having 2 attributes) was that, for Table 2, there was one edge and one vertex attribute but there were only two edge attributes corresponding to Table 3. Overall, performance of all methods was better in Table 2 than that shown in Table 3.

Based on above experimental result, some conclusion can be drawn.

- 1) For low noise conditions, and with simple adjacency matrices or weighted graphs, SM is superior to LSM.
- 2) For attributed graph matching ($r, s > 1$) the LSM is superior to SM. Further, the decomposition method used in conjunction with LSM consistently improves performance under all conditions.
- 3) Increasing attributes of vertices is more helpful in improving performance than increasing the number of edge attributes.
- 4) When the additive noise is large all three methods have poor performance, suggesting the non-linear method [3], should be used.

Table 1. Performance comparison of LSM and SM, $(r, s) = (1, 0)$, $(n', n) = (10, 10)$, $(30, 30)$, $(50, 50)$. See Fig. 2 for definitions of terms.

ε	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
LSM(10)	63.7	49.8	35.5	29.0	24.5	22.5	19.9	18.2
SM(10)	84.5	55.5	42.5	23.6	19.1	17.7	14.2	12.4
LSM(30)	33.7	18.5	13.6	10.8	9.6	8.3	7.9	6.3
SM(30)	72.8	37.2	15.9	8.9	5.9	5.6	5.3	4.6
LSM(50)	21.6	12.1	8.7	6.9	6.2	5.1	4.8	4.3
SM(50)	55.8	25.5	7.9	4.9	3.8	2.9	2.8	2.5

Table 2. Performance comparison of LSM and SM, $(r, s) = (1, 1)$, $(n', n) = (10, 10)$, $(30, 30)$, $(50, 50)$. See Fig. 2 for definitions of terms.

ε	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
LSM(10)	94.9	84.2	69.9	58.1	50.0	41.5	34.5	33.5
SM(10)	93.0	68.5	48.2	30.9	23.8	21.4	19.6	16.2
LSM(30)	85.8	60.6	39.9	27.5	20.7	17.9	14.4	11.1
SM(30)	84.0	23.9	14.1	8.0	6.6	6.2	5.0	4.3
LSM(50)	77.7	45.4	28.1	19.1	14.1	11.1	8.9	8.3
SM(50)	69.8	13.3	8.6	5.0	3.5	3.1	2.8	2.0

Table 3. Performance comparison of LSM and SM, $(r, s) = (2, 0)$, $(n', n) = (10, 10)$, $(30, 30)$, $(50, 50)$. (See Fig. 2 for definitions of terms).

ε	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
LSM(10)	90.0	74.1	56.1	48.0	39.1	30.8	26.1	25.1
SM(10)	91.3	56.4	31.4	25.0	18.1	16.0	15.0	13.5
LSM(30)	76.4	46.2	32.5	22.5	17.4	14.6	12.0	10.8
SM(30)	82.1	20.9	9.9	6.2	4.8	4.4	3.8	3.5
LSM(50)	66.3	33.9	21.9	14.5	11.9	9.3	7.4	6.4
SM(50)	65.4	11.9	4.7	3.4	3.04	2.8	2.7	2.1

6 Conclusions

In this paper, two main linear algebraic methods for attributed graph matching, LSM and SM have been compared along with a new graph decomposition model, combined with LSM, for attributed graph matching. For most cases, the LSM is superior to SM most probably due to the fact that the LSM uses more information than the SM. However, when there are few constraints such as in attributed or weighted graphs, LSM performance is inferior to the SM. For this reason SM would be the preferred algorithm for matching graphs defined only by their adjacency matrices – the more commonly used graph data model for recent SM methods. But, the SM is quite sensitive to noise. However, when there is little noise, its performance is quite good. In so far as LSM methods are concerned when a graph decomposition model is combined with the LSM, generally speaking, performance along all dimensions improves. However, when noise and/or vertices numbers are large even LSM performance is not satisfactory – suggesting that non-linear optimization, search methods, may provide better solutions.

References

1. van Wyk, B.J., Van Wyk, M. A.: Kronecker product graph matching. *Pattern Recognition* 36 9(2003), 2019-2030.
2. El-Sonbaty, Y., Ismail, M.A.: A new algorithm for subgraph optimal isomorphism. *Pattern Recognition* 31 2 (1998), 205–218
3. Gold, S., Rangarajan, A.: A graduated assignment algorithm for graph matching. *IEEE TPAMI* 18 4 (1996), 377–388
4. Umeyama, S.: An Eigen Decomposition Approach to Weighted Graph Matching Problems. *IEEE TPAMI* 10 5 (1988), 695-703
5. Scott, G. L., Longuet-Higgins, H.C.: An algorithm for associating the features for two patterns. *Proc Roy Soc Lond*, Vol B244 (1991), 21-26
6. Shapiro, L., Brady, J.M.: Feature-based Correspondence: an eigenvector approach, *Image & Vision Computing*, 10 5 (1992), 283-288
7. Luo, B., Wilson, R.C., Hancock, E.R.: Spectral embedding of graphs. *Pattern Recognition* 36 9 (2003), 2213–2230.
8. Kosinov, S. and Caelli, T.: Inexact Multisubgraph matching using Graph Eigenspace and Clustering Models. 9th International Workshop on Structural and Syntactic Pattern: SSPR2002. In: Caelli, T., Amin, A., Duin, R., Kamel, M., de Ritter, D. (Eds.), *Structural, Syntactic and Statistical Pattern Recognition, Lecture Notes in Computer Science*, LNCS 2396, Springer, 133-142.
9. Clark, J., Holton, D.A.: *A First Look at Graph Theory*. World Scientific Publishing, Singapore (1991).
10. Yamada, S., Hasai, T.: An efficient algorithm for the linear assignment problem. *Elect. Comm. Japan* 73 12 (1990), 28–36 Part 3.
11. Grewe, L., Kak, A.C.: Interactive learning of a multiple-attribute hash table classifier for fast object recognition. *CVIU* 61 3 (1995), 387-416.