Graph Clustering using Symmetric Polynomials and Local Linear Embedding

Richard Wilson, Xiao Bai and Edwin R. Hancock
Department of Computer Science University of York UK

Abstract

Although graph structures have proved useful in high level vision for object recognition and matching, they can prove computationally cumbersome because of the need to establish reliable correspondences between nodes. Hence, standard pattern recognition techniques can not be easily applied to graphs since feature vectors and not easily contructed. To overcome this problem, in this paper we turn to the spectral matrix. We show how the elements of this matrix can be used to construct symmetric polynomials that are permutation invariants. The co-efficients of these polynomials can be used as graph-features which can be encoded in a vectorial manner. We demonstrate that these vectors can be embedded in a low dimensional space using locally linear embedding, and that the embedding results in well defined graph clusters.

1 Introduction

Graph structures have proved important in high level vision since they can be used to represent structural and relational arrangements of objects in a scene. The key problem in utilising graph representations lies in measuring their structural similarity. In general, the nodes of a graph are not ordered or labelled, and the node correspondence problem must be solved before structural similarity to be assessed. In other words, a mapping must be found between the nodes of the graphs being compared. This is equivalent to finding a permutation of the nodes. For noisy graphs (those which are subject to structural differences) this problem is thought to be NP-hard. There are a number of ways in which this problem may be tackled. Many authors have employed the concept of graph edit distance. The idea here is to perform elementary editing operations on a graph, such as edge or node insertion and deletion, to make pairs of graphs isomorphic. Each operation has an associated 'cost', and the minimum total cost of the set of edit operations can be used to gauge the similarity of the graphs. For example, Fu et al [5, 11] have computed similarities using separate edit costs for relabeling, insertion and deletion on both nodes and edges. A search is necessary to locate the set of operations which have minimal cost. More recently, Bunke [2] has established a relationship between the minimum graph edit distance and the size of the maximum common subgraph. Torsello and Hancock [1] have exploited this relationship to cast the problem into a continuous optimisation framework. Shapiro and Haralick [8] have exploited a similarity measure based on the number of consistent structural relationships in pairs of graphs. Again a search is exploited to locate the best correspondence mapping between the nodes.

BMVC 2003 doi:10.5244/C.17.34

There are two main conclusions to draw from this brief review of the literature. Firstly, the calculation of graph similarity requires the solution of the correspondence problem as a prerequisite. This problem is typically NP-hard and therefore computationally expensive. Secondly, there is no clear consensus on how to compute the similarity of graphs. Hence, graphs can not be manipulated as easily as pattern vectors, where the order and identity of components features is fixed and known, and the computation of similarity typically relies on a simple inner product between the vectors. In fact the non-vectorial nature of graphs has meant that pattern analysis tasks such as clustering, the analysis of variance and dimensionality reduction can not be applied to graphs. This in turn has meant that machine learning in the graph-domain has proved to be an elusive problem.

In this paper we investigate whether it is possible to construct pattern-vectors for graphs in a manner which does not require the solution of the correspondence problem and which allows similarity can be easily assessed. The adopted approach is based on spectral graph theory, which is a branch of mathematics that. is concerned with characterising the structural properties of graphs using the eigenvectors of the adjacency matrix or the closely related Laplacian matrix (the degree matrix minus the adjacency matrix) [3]. One of the well known successes of spectral graph theory in computer vision is the use of eigenvector methods for grouping via pairwise clustering. Examples include Shi and Malik's [9] iterative normalised cut method which uses the Fiedler (i.e. second) eigenvector for image segmentation and Sarkar and Boyer's use of the leading eigenvector of the weighted adjacency matrix [6] for line-grouping. Graph spectral methods have also been used to correspondence analysis. For instance, Scott and Longuet-Higgins[7] have used the eigenvectors of a point-affinity matrix to perform alignment and correspondences. Umeyama[12] has developed a method for finding the permutation matrix which best matches pairs of weighted graphs, using a singular value decomposition of the adjacency matrices.

Although graph-spectral methods have proved effective in computer vision, the existing methods have not made full use of the available spectral representation. In other words, most of the methods described above are based on the use of a single eigenvector. In this paper, on the other hand, we describe a method for constructing spectral features which are permutation invariants. To construct these invariants we use symmetric polynomials. The arguements of the polynomials are the elements of the spectral matrix. We use the co-efficients of the symmetric polynmials to construct graph pattern-vectors. With the vectors to hand, we explore how they may be used to construct pattern spaces for sets of graphs. We explore two different approaches. The first of these is to apply principal components analysis to the covariance matrix for the vectors. The second approach is to apply locally linear embedding to the vectors. This is a variant of PCA that modifies the distortion measure using a nearest neighbour criterion. We demonstrate that this latter method gives the best graph-clusters.

2 Spectral graph representation

Consider the undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node-set $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ and edge-set $\mathcal{E} = \{e_0, e_1, \ldots, e_m\} \subset \mathcal{V} \times \mathcal{V}$. The adjacency matrix $\mathbf{C}$ for the graph $\mathcal{G}$ is the $n \times n$ symmetric matrix with elements

$$C_{ab} = \begin{cases} 1 & \text{if } (v_a, v_b) \in \mathcal{E} \\ 0 & \text{otherwise} \end{cases}$$

In general the task of comparing two such graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ involves finding a correspondence mapping between the nodes of the two graphs, $f : \mathcal{V}_1 \cup \phi \leftrightarrow \mathcal{V}_1 \cup \phi$ where 'ϕ' represents a null match or dummy node which may be used the account for structural differences between the two graphs. The correspondence mapping problem may be posed as that of finding the permutation of nodes in the graph $\mathcal{G}_2$ which places them in the same order as those in the graph $\mathcal{G}_1$. This permutation can be applied to the adjacency matrix of graph $\mathcal{G}_2$ so that it may be compared with that for graph $\mathcal{G}_1$. If the graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ are isomorphic then the permutation matrix P satisfies the condition $\mathbf{C}_1 = \mathbf{P}\mathbf{C}_2\mathbf{P}^T$. When the graphs are dissimilar, then the optimal permutation matrix can be found by minimising the difference between $\mathbf{C}_1$ and $\mathbf{P}\mathbf{C}_2\mathbf{P}^T$. Graph spectral techniques have been used to solve this problem. For instancem Umeyama[12] has shown how singular value decomposition can be used to find the permutation matrix which satisfies the condition $P^* = \arg\min_P ||C_1 - PC_2P^T||$, i.e. which minimises the Froebenius norm. In some applications, especially in structural chemistry, eigenvalues have also been used to compare the structural similarity of different graphs. However, although the eigenvalue spectrum is a permutation invariant, it represents only a fraction of the information residing in the eigensystem of the adjacency matrix.

2.1 The graph spectral matrix

We commence our discussion of graph-spectra by considering how to construct the adjacency or connection matrix. The graphs considered here have no self-connections and hence $C_{ii} = 0$. However, if the diagoanl elements are set to zero in this way then the spectral decomposition of the adjacency matrix will have many negative eigenvalues. This problem is generally solved by setting the diagonal element equal to the degree of the corresponding node, i.e. $C_{ii} = \sum_{j \neq i} C_{ij}$. The matrix $\mathbf{C}$ is then positive definite, and the eigenvalues are hance all either positive or zero.

The spectral matrix for a graph is found by performing an eigenvector decomposition of the the adjacency matrix $\mathbf{C}$, i.e. $\mathbf{C} = \sum_{i=1}^{n} \lambda_i \mathbf{e}_i \mathbf{e}_i^T$ where λ_i is the ith eigenvalue and $\mathbf{e}_i$ is the corresponding eigenvector of the symmetric matrix $\mathbf{C}$. The eigenvectors are normalized so that they have unit length i.e. $\mathbf{e}_i^T \mathbf{e}_i = 1$. With the eigenvectors to hand the spectral matrix is given by $\Phi = (\sqrt{\lambda_1}\mathbf{e}_1^T, \sqrt{\lambda_2}\mathbf{e}_2^T, ..., \sqrt{\lambda_n}\mathbf{e}_n^T)^T$. The matrix Φ is a complete representation of the graph in the sense that, providing there are no negative eigenvalues, we can reconstruct the original adjacency matrix using the equation $\mathbf{C} = \Phi^T\Phi$.

2.2 Uniqueness of the spectral matrix

Theorem 1 *Matrix Φ is a unique representation of $\mathbf{C}$ iff all n eigenvalues are distinct.*

This follows directly from the fact that there are n distinct eigenvectors when the eigenvalues are all distinct. When an eigenvalue is repeated, there exists a subspace spanned by the eigenvectors of the degenerate eigenvalues in which all vectors are also eigenvectors of C. In other words, if there are two equal eigenvalues, $\lambda_a = \lambda_b$, then $\mathbf{e}'$ is also an eigenvector of $\mathbf{C}$, where

$$\mathbf{e}' = \alpha\mathbf{e}_a + \beta\mathbf{e}_b, \ \alpha^2 + \beta^2 = 1$$

The condition $\alpha^2 + \beta^2 = 1$ maintains the normalization of the eigenvector. In this case there is continuum of representations.

If the eigenvalues are distinct, then the graph is such that every automorphism, excepting the identity, is of order 2[4]. In this case, the isomorphism problem is known to be easy. In the following, we will refer to such graphs as *simple*.

3 Node permutations and invariants

The topology of a graph is invariant under permutations of the node labels. However, the adjacency matrix is modified by the permutation since the rows and columns are indexed by the node-order. Hence, if we relabel the nodes, the adjacency matrix undergoes a permutation of both rows and columns. Let the matrix $\mathbf{P}$ be the permutation matrix representing the change in node order. The permuuted adjacency matrix is given by $\mathbf{C}' = \mathbf{PCP}^T$. There is hence a family of adjacency matrices which can be transformed into one-another using a permutation matrix. The spectral matrix is also modified by permutations.

Theorem 2 *Matrix Φ is unique up to a permutation of the columns for simple graphs.*

Let $\mathbf{C}$ be the adjacency matrix of a graph $\mathscr{G}$ and let $\mathbf{C}' = \mathbf{PCP}^T$ be the adjacency matrix obtained by the permutation P. Further, let $\mathbf{e}$ be a normalised eigenvector of $\mathbf{C}$ with associated eigenvalue λ, and let $\mathbf{e}' = \mathbf{Pe}$. With these ingredients, we have that

$$\mathbf{C}'\mathbf{e}' = \mathbf{PCP}^T\mathbf{Pe} = \mathbf{PCe} = \lambda\mathbf{e}' \tag{1}$$

Hence, $\mathbf{e}'$ is an eigenvector of $\mathbf{C}'$ with associated eigenvalue λ. As a result, we can write $\Phi' = \Phi P^T$. Direct comparison of the spectral matrices for different graphs is hence not possible because of the unknown permutation.

The eigenvalues of the adjacency matrix have been used as a compact spectral representation for comparing graphs. The eigenvalues can be recovered from the spectral matrix using the identity

$$\lambda_i = \sum_j \Phi_{ij}^2$$

The expression $\sum_j \Phi_{ij}^2$ is infact a *symmetric polynomial* in the components of eigenvector $\mathbf{e}_i$. A symmetric polynomial is invariant under permutation of the variable indicies.

The eigenvalue is one example of an infinite family of symmetric polynomials which can be defined on the components of the eigenvectors, i.e. the elements of the spectral matric. However, there is a special set of these polynomials, referred to as the *elementary symmetric polynomials*($\mathscr{S}$) that form a basis set for symmetric polynomials. In other words, any symmetric polynomial can itself be expressed as a polynomial function of the elementary symmetric polynomials belonging to the set $\mathscr{S}$.

We therefore turn our attention to the set of symmetric polynomials. For a set of variables $\{x_1, x_2 \ldots x_n\}$ they can be defined as

$$S_1(x_1,\ldots x_n) = \sum_{i=1}^{n} x_i \quad S_r(x_1,\ldots x_n) = \sum_{i_1<i_2<\ldots<i_r} x_{i_1}x_{i_2}\ldots x_{i_r} \quad S_n(x_1,\ldots x_n) = \prod_{i=1}^{n} x_i \tag{2}$$

The polynomial functions

$$P_1(x_1,\ldots x_n) = \sum_{i=1}^{n} x_i \quad P_n(x_1,\ldots x_n) = \sum_{i=1}^{n} x_i^n \tag{3}$$

also form a basis set over the set of symmetric polynomials. As a consequence, any function which is invariant to permutation of the variable indicies and that can be expanded as a Taylor series, can be expressed in terms of one of these sets of polynomials.

The two sets of polynomials are related to one another by the Newton-Girard formula:

$$S_r = \frac{(-1)^{r+1}}{r} \sum_{k=1}^{r} (-1)^{k+r} P_k S_{r-k} \tag{4}$$

In this paper, we intend to use the polynomials to construct invariants from the elements of the spectral matrix, The elementary simple polynomials can hence provide spectral "features" which are invariant under node permutations of the nodes in a graph.

4 Feature distributions

While the elementary symmetric polynomials provide spectral features which are invariant to permutations, they are not suitable as a representation for gauging the difference between graphs. The distribution of S_r for large r is highly non-Gaussian with a dominant tail because of the product terms appearing in the higher order polynomials. In order to make the distribution more tractable, it is convenient to take logs. Furthermore, the components of Φ may be zero, which will lead to loss of information for certain members of the set $\mathscr{S}$. As an example, if any component of $\vec{e}_i$ is zero, then $S_n = 0$. If we wish to take logs, the condition $S_r > 0 \ \forall r$ must hold. We therefore perform a component transform $\Phi_{ij} \rightarrow \varepsilon + |\Phi_{ij}|$. It is convenient to choose ε so that the mean-value of the components is 1. Hence, we construct the following matrix from the symmetric polynomials

$$F_{ij} = \ln S_j(\Phi_{i,1}, \Phi_{i,2}, \ldots, \Phi_{i,n}) \tag{5}$$

where $1 \leq i \leq n, 1 \leq j \leq n$. The columns of this matrix are stacked to form a long-vector B.

5 Pattern Spaces

We would like to explore how the feature-vectors extracted from the symmetric polynomoals can be used to construct pattern-spaces for sets of graphs. We explore two approaches. The first of these in principal components analysis. The second method is locally linear embedding [10].

5.1 Principal Components Analysis

Suppose that we have a set of graphs $\mathscr{G}_1$, $\mathscr{G}_\in$,..., $\mathscr{G}_M$ with associated long-vectors of spectral features B_1, B_2,....., B_M. The N different image vectors are arranged in order as the columns of the matrix $S = \left[B_1|B_2|\ldots|B_i|\ldots|B_N\right]$. For the set of spectral feature vectors, the covariance matrix is $\Sigma = SS^T$ and the eigendecomposition of the matrix is $\Sigma = \sum_{i=1}^{N} \lambda_i \vec{u}_i \vec{u}_i^T$, where λ_i are the eigenvalues and $\vec{u}_i$ are the corresponding eigenvectors of Σ. We project the spectral feature vectors onto the space spanned by the leading d eigenvectors. Let $T = (\vec{u}_1|\vec{u}_2|\ldots|\vec{u}_d)^T$ be the matrix with the leading eigenvectors of Σ as rows. The projection of the feature vector B_j onto this eigenspace is $z_j = TB_j$, Hence the graph G_j is represented by a d-component vector $\vec{z}_j$ in the eigenspace.

5.2 Locally Linear Embedding

The second dimension reduction method that we use is locally linear embedding (LLE) [10]. For each pair of graphs $\mathcal{G}_i$ and $\mathcal{G}_j$, we compute the squared Euclidean distance $d^2_{i,j} = (B_i - B_j)^T(B_i - B_j)$. For the graph B_i we find the index set K_i of the set of graphs which are the K-nearest neighbours according to the Euclidean distance. Using the K-nearest neighbours, we construct a $M \times M$ weight matrix W whose elements are defined as follows

$$W_{i,j} = \begin{cases} \frac{1}{K} & \text{if } j \in K_i \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The distortion of the data is guaged by the measure

$$E(W) = \sum_{i=1}^{M}\sum_{j=1}^{M}(B_i - WB_j)^T(B_i - WB_j) \tag{7}$$

The aim in locally linear embedding is to find a projection of the vectors B_i into a d-dimensional subspace that minimises the distortion measure in this subspace. This projection may be found from the eigenvectors of the matrix

$$D = (I - W)^T(I - W) \tag{8}$$

The eigenvector $\vec{u}_1$ corresponding to the smallest eigenvalue of M is the "all-ones" vector, and conveys no information. The eigenvectors $\vec{u}_2$, $\vec{u}_3$,....$\vec{u}_{d+1}$ corresponding to the next d eigenvalues of increasing size are the basis vectors of the required projection. Following the approach adopted with PCA we construct the projection matrix $T = (\vec{u}_2|\vec{u}_3|\ldots|\vec{u}_{d+1})^T$ to compute the d-vectors $z_j = TB_j$ for the purposes of visualisation.

6 Experiments

There are two aspects to the experimental evaluation of the techniques reported in this paper. We commence with a study on synthetic data aimed at evaluating the ability of the spectral features to distinguish between graphs under controlled structural error. The second part the study focusses on real world data and assesses whether the spectral feature vectors can be embedded in a pattern space that reveals cluster-structure.

6.1 Synthetic Data

We commence by examining the ability of the spectral feature set and distance measure to separate both structurally related and unrelated graphs. This study utilises random graphs which consist of 30 nodes and 140 edges. The edges are generated by connecting random pairs of nodes. Structurally related graphs are generated from an original graph by an edit operation of either deleting an edge or inserting a new random edge. Each of these operations is assigned a cost 1, and therefore a graph with a deleted edge has an edit distance 1 to the original. In the left-hand panel of Figure 1 we have plotted the distance(ε) distributions of an edited graph of edit distance 1 and the set of random graphs with respect to a reference graph. The edited graph is derived from the reference graph by the removal of a random edge. The results in table 1 demonstrate the performance

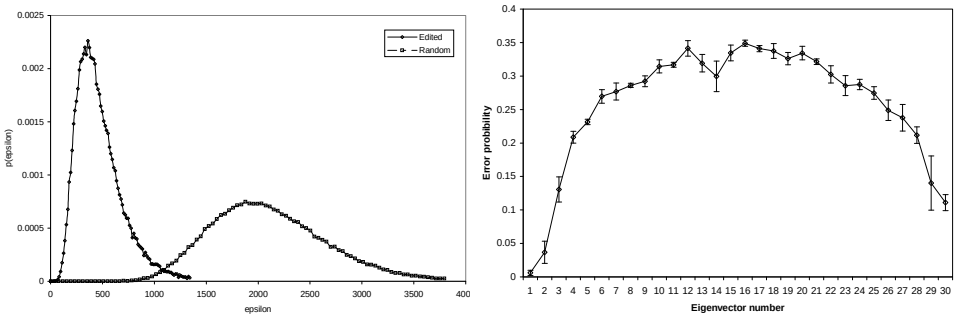

Figure 1: Distributions of distance to edited (left) and discriminating power of individual eigenvectors (right).

Edit distance	1	2	3	4	5
Confusion prob.	0.015	0.047	0.098	0.144	0.179
Edit distance	6	7	8	9	10
Confusion prob.	0.209	0.234	0.262	0.283	0.289

Table 1: Performance of feature set for edited graphs

under different levels of corruption. The method works well for very similar graphs, but degrades at higher levels of corruption.

In the right hand panel of Figure 1 we illustrate the relative ability of each eigenvector to discriminate between the random and 1-edit classes of graphs. Here only one eigenvector is used in the distance measure. Eigenvector-n denotes the eigenvector with the n^{th} largest eigenvalue. The plot demonstrates that the leading and the last few eigenvectors are the most important in defining the structure of the graph.

At first sight this suggests that only a few of the eigenvectors need be used to characterise the graph. However, as Table 2 reveals this is only the case for small edit distances. As the edit distance increases, then more of the eigenvectors become important.

To take this study one step further, we create three clusters of graphs from three reference graphs by perturbing them by edit operations. The distances between these graphs are then used to embed the graphs in a two dimensional space using multi-dimensional scaling. The procedure is as follows. Three seed graphs are generated at random. These seed graphs are used to generate samples by deleting an edge at random from the graph. The distance matrix is then constructed, which has elements d_{ij}, the distance between

	Edit distance			
Eigenvectors	1	2	3	4
First only	0.003	0.100	0.144	0.211
Start 5 and end 5	0.043	0.088	0.132	0.171
All	0.015	0.047	0.098	0.144

Table 2: Confusion probabilities of different eigenvector sets and edit distances

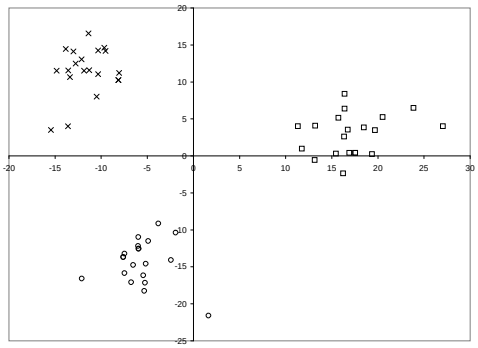

Figure 2: MDS distribution of graphs from three random sets.

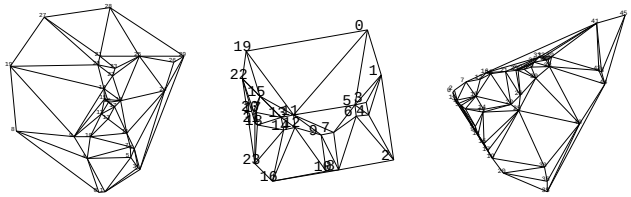

Figure 3: Example graphs from 3 different model sets; left: ‘CMU’, centre: ‘model’, right ‘chalet’

graphs i and j. Finally, the MDS technique is used to visualise the distribution of graphs in a 2 dimensional space. The results are shown in Figure 2. The graphs form good clusters and are well separated from one another.

6.2 Real World Data

We experiment with three sequences of 2D views of model houses under gradually varying viewpoint. From each image in each view sequence, we extract corner features. We use the extracted corner points to construct Delaunay graphs (examples are shown in Figure 3). Ideally, we we expect the graphs belonging to the same sequence to be close in pattern-space, while the different sequences should be well separated.

In the top row Figure 4 we show the results of applying PCA (left) and LLE (right) to the spectral feature vectors. In the plots, the images in the three sequences are indicated by the numbers from 1-10, from 11-20 and 21-30. The main feature to note from the two plots is that LLE produces an embedding in which the three image sequences are better clustered than with PCA. However, in the case of PCA, the different graphs describe a smooth trajectory, and hence the method may prove useful for view interpolation. In the bottom row, we show the results of applying PCA (left) and LLE (right) to a conventional spectral representation. The simple representation chosen is the vector of eigenvalues. In both cases the cluster structure appears pooer than when the spectral feature-vectors are used.

To investigate the behaviour of LLE when used in conjunction with the spectral feature-vectors, in Figure 5 we show the effect of varying some of the parameters of the method. In the top row of the figure, we show the effect of varying the number of nearest neighbours. In the bottom row, we show the effect of varying the number of components of the spectral feature vector. The main features to note from these plots are that the best clusters emerge when k=4, and that little improvement is obtained once 30% of the available components of the spectral feature vector are used.

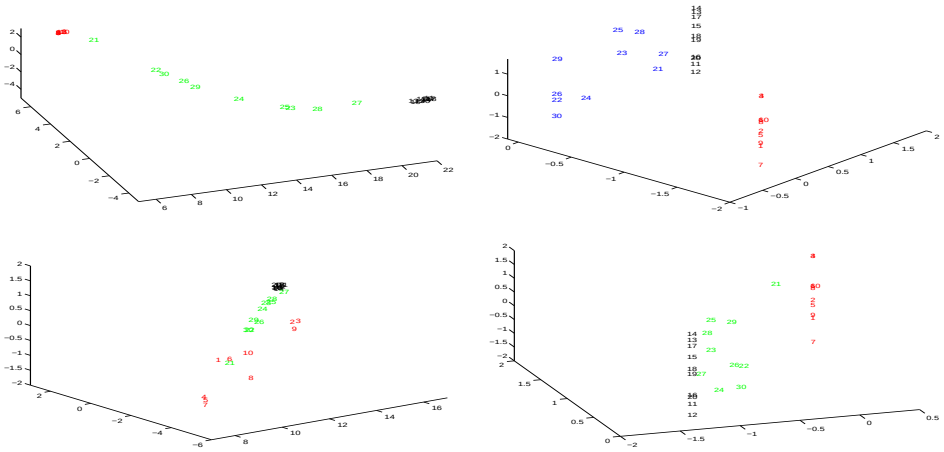

Figure 4: Comparing PCA and LLE for the spectral feature vectors and the eigenvalue spectrum.

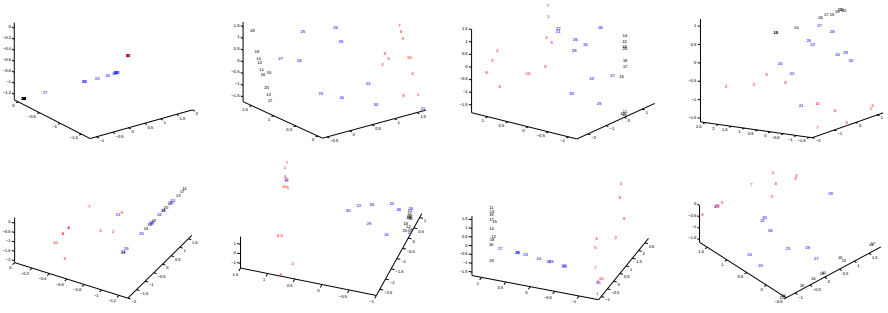

Figure 5: The effects of k-nearest neighbours and feature-vector length on LLE.

7 Conclusions

In this paper we have shown how graphs can be converted into pattern vectors by utilising the spectral decomposition and basis sets of symmetric polynomials. We have shown that

these feature vectors are complete, unique and continuous.

We then investigate how to embed the vectors in a pattern space, suitable for clustering the graphs. Here we explore two alternatives. The first of these is PCA. The second is locally linear embedding. This is a variant of PCA which uses a nearest neighbour criterion to modify the computation of distortion. Results show that LLE gives the best clusters.

References

[1] A.Torsello and E.R.Hancock. Efficiently computing weighted tree edit distance using relaxation labeling. *Lecture Notes in Computer Science*, 2134:438–453, 2001.

[2] H. Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18:689–694, 1997.

[3] F. R. K. Chung. *Spectral Graph Theory*. AMS, 1997.

[4] A. Mowshowitz. The group of a graph whose adjacency matrix has all distinct eigenvalues. *Proof Techniques in Graph Theory, Academic Press, New York*, pages 109–110, 1969.

[5] A. Sanfeliu and K. S. Fu. A distance measure between attributed relational graphs for pattern-recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 13(3):353–362, 1983.

[6] S. Sarkar and K. L. Boyer. Preceptual organization in computer vision. *IEEE Trans. Systems, Man and Cybernetics*, 23:382–399, 1993.

[7] G. Scott and H. Longuet-Higgins. An algorithm for associating the features of two images. *Proceedings of the Royal Society of London Series B-Biological*, 244:21–26, 1991.

[8] L. G. Shapiro and R. M. Haralick. Structural descriptions and inexact matching. *IEEE Transations on Pattern Analysis and Machine Intelligence*, 3(5):504–519, 1981.

[9] J. Shi and J. Malik. Normalized cuts and image segmentation. *CVPR*, pages 731–737, 1997.

[10] S.T.Roweis and L.K.Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[11] W. H. Tsai and K. S. Fu. Subgraph error-correcting isomorphisms for syntactic pattern-recognition. *IEEE Transations on Systems, Man and Cybernetics*, 13(1):48–62, 1983.

[12] S. Umeyama. An eigendecompostion approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.