A Probabilistic Framework for Graph Clustering

Bin Luo, Antonio Robles-Kelly*, Andrea Torsello, Richard C. Wilson and Edwin R. Hancock
Department of Computer Science
University of York
York Y01 5DD, UK.
{atorsell, erh}@cs.york.ac.uk

Abstract

This paper describes a probabilistic framework for graph-clustering. We commence from a set of pairwise distances between graph-structures. From this set of distances, we use a mixture model to characterize the pairwise affinity of the different graphs. We present an EM-like algorithm for clustering the graphs by iteratively updating the elements of the affinity matrix. In the M-step we applying eigendcomposition to the affinity matrix to locate the principal clusters. In the M-step we update the affinity probabilities. We apply the resulting unsupervised clustering algorithm to two practical problems. The first of these involves locating shape-categories using shock trees extracted from 2D silhouettes. The second problem involves finding the view structure of a polyhedral object using the Delaunay triangulation of corner features.

1. Introduction

Graph clustering is an important, yet relatively under-researched topic in machine learning [1, 2, 3]. The importance of the topic stems from the fact that it is an important tool for learning the class-structure of data abstracted in terms of relational graphs. Problems of this sort are posed by a multitude of unsupervised learning tasks in knowledge engineering, pattern recognition and computer vision. The process can be used to structure large data-bases of relational models [4] or to learn equivalence classes. One of the reasons for limited progress in this area has been the lack of algorithms suitable for clustering relational structures. In particular, the problem has proved elusive to conventional central clustering techniques. The reason for this is that it has proved difficult to define what is meant by the mean or representative graph for each cluster. However, Munger, Bunke and Jiang [5] have recently taken some important steps in this direction by developing a genetic algorithm for searching for median graphs. A more fruitful avenue of investigation may be to pose the problem as pairwise clustering. This requires only that a set of pairwise distances between graphs be supplied. The clusters are located by identifying sets of graphs that have strong mutual pairwise affinities. There is therefore no need to explicitly identify an representative (mean, mode or median) graph for each cluster. Unfortunately, the literature on pairwise clustering is much less developed than that on central clustering.

When posed in a pairwise setting, the graph-clustering problem requires two computational ingredients. The first of these is a distance measure between relational structures. The second is a means of performing pairwise clustering on the distance measure. There are several distance measures available in the literature. For instance, in the classical pattern recognition literature, Haralick and Shapiro [6] have described a relational distance measure between structural descriptions, while Sanfeliu and Fu [7] have extended the concept of edit distance from strings to graphs. There have also been attempts to use an information theoretic approach. Here Wong and You [8] have computed the entropy for random graphs, while Boyer and Kak [9] have used mutual information. More recently, Christmas, Kittler and Petrou [10], and Wilson and Hancock [11] have developed probabilistic measures of graph-similarity. Turning our attention to pairwise clustering, there are several possible routes available. The simplest is to transform the problem into a central clustering problem. For instance, it is possible to embed the set of pairwise distances in a Euclidean space using a technique such as multi-dimensional scaling and to apply central clustering to the resulting embedding. The second approach is to use a graph-based method [12] to induce a classification tree on the data. Finally, there are mean-field methods which can be used to iteratively compute cluster-membership weights [13]. These methods require that the number of pairwise clusters be known *a priori*.

In this paper we use a maximum likelihood framework for graph clustering recently developed by Robles-Kelly and Hancock[18]. The problem is posed as one of pairwise

*Supported by CONACYT, under grant No. 146475/151752.

0-7695-1272-0/01 $10.00 © 2001 IEEE

clustering which is parameterized using two sets of indicator variables. The first of these are cluster membership variables which indicate to which cluster a graph belongs. The second set of variables are affinity weights which convey the strength of the similarity relations between pairs of graphs belonging to the same cluster. Our clustering algorithm is an iterative one in which both sets of indicator variables are updated using a process which bears similarities with the EM algorithm.

We focus on two applications of the graph clustering method. The first of these is concerned with the unsupervised learning of shape-categories. This involves the abstraction of 2D binary shapes using shock-trees. Commencing from a data-base of silhouettes, we extract the Hamilton-Jacobi skeleton and locate the shocks which correspond to singularities in the evolution of the object boundary under the eikonal equation. We compute the similarity of the shapes using weighted tree edit distance. The second application involves identifying the view-structure of 3D objects from 2D images. Here the graphs are Delaunay triangulations of corner features. The distance measure between the graphs furnished by a recently reported matrix factorization method [21].

2. Graph Affinity

We pose the problem of graph-clustering as that of finding pairwise clusters in the distribution of graph distance. The process of pairwise clustering is somewhat different to the more familiar one of central clustering. Whereas central clustering aims to characterize cluster-membership using the cluster mean and variance, in pairwise clustering it is the relational similarity of pairs of objects which are used to establish cluster membership. Although less well studied than central clustering, there has recently been renewed interest in pairwise clustering aimed at placing the method on a more principled footing using techniques such as mean-field annealing [13].

To commence, we require some formalism. We are interested in grouping a set of graphs $\mathcal{G} = \{G_1, ..., G_{|M|}\}$ whose index set is M. The set of graphs is characterized using a matrix of pairwise similarity weights. The elements of this weight matrix are computed using a graph distance $d_{i,j}$ between the graphs indexed i and j. The methods used to compute this distance are application dependent and are described in Section 4 of this paper.

We adopt the following picture of the graph clustering process. The picture revolves around the idea that the graphs can be embedded in a n-dimensional space $\mathcal{R}^n$. Here we treat the embedding space as a latent representation. Hence we are not concerned with a specific embedding procedure. However, a number of concrete possibilities exist. For instance, features could be extracted from the graph adjacency structure and subjected to principal components analysis, or the pattern of pairwise distances could be subjected to multidimensional scaling. Hence, each graph becomes a point in the embedding space. We assume that for each distinct cluster of graphs the embedded position vectors follow a spherically symmetric Gaussian distribution. For the cluster with index ω, the covariance matrix is $\sigma_\omega I_n$ where n is the $n \times n$ identity matrix. Suppose that $x_{i\omega}$ and $x_{j\omega}$ represent the embedded position vectors for the graphs G_i and G_j, and that the graphs both belong to the cluster indexed ω. The difference in position between the graphs, i.e. $x_{i\omega} - x_{j\omega}$ will be drawn from the normal distribution $\mathcal{N}(0, 4\sigma_\omega^2 I)$. As a result the distance measure

$$\left\| \frac{x_{i\omega} - x_{j\omega}}{2\sigma_\omega} \right\| = \frac{d_{ij}^2}{4\sigma_\omega^2} \approx \chi_n^2. \tag{1}$$

will follow a χ^2 distribution with n degrees of freedom.

Given a distance d_{ij} between two points i and j, we can, estimate the probability that the two points belong to the same cluster ω', using the χ^2 distribution provided that we know the cluster variance $\sigma_{\omega'}^2$. The estimated probability is:

$$P\{i \text{ and } j \in \omega'\} = P\left\{ \chi_n^2 > \frac{d_{ij}^2}{4\sigma_{\omega'}^2} \right\}. \tag{2}$$

Using this simple model, we can define the similarity matrix W setting its coefficients W_{ij} to the probability that the graphs i and j belong to the same cluster. In other words:

$$W_{ij} = P\left\{ \chi_n^2 > \frac{d_{ij}^2}{4\sigma_{\omega'}^2} \right\}. \tag{3}$$

3. Clustering

The aim in graph-clustering is to update a set of similarity weights which partition the set of graphs into disjoint subsets. Let S_ω represent the index-set of the cluster of graphs indexed ω. Since the different clusters are disjoint $S_{\omega'} \cap S_{\omega''} = \emptyset$ if $\omega' \neq \omega''$.

In this paper we are interested using matrix factorization methods to locate the clusters. One way of viewing this is to search for the permutation matrix which re-orders the elements of W into non-overlapping blocks. However, when the elements of the matrix W are not binary in nature, then this is not a straightforward task. However, Sarkar and Boyer [14] have shown how the same-sign eigenvectors of the matrix of similarity-weights can be used to for clustering. Using the Rayleigh-Ritz theorem, they observe that the scalar quantity $\mathbf{v}^t W \mathbf{v}$, where W is the weighted adjacency matrix, is maximized when $\mathbf{v}$ is the leading eigenvector of W. Moreover, each of the subdominant eigenvectors corresponds to a disjoint cluster. We confine our attention to

the same-sign eigenvectors (i.e. those whose corresponding eigenvalues are real and positive, and whose components are either all positive or are all negative in sign). If a component of a same-sign eigenvector is non-zero, then the corresponding node belongs to the cluster associated with the eigen-modes of the similarity weight matrix. The eigenvalues $\lambda_1, \lambda_2 ....$ of W are the solutions of the equation $|W - \lambda I| = 0$ where I is the $|M| \times |M|$ identity matrix. The corresponding eigenvectors $\mathbf{v}_{\lambda_1}, \mathbf{v}_{\lambda_2}, ...$ are found by solving the equation $W\mathbf{v}_{\lambda_i} = \lambda_i \mathbf{v}_{\lambda_i}$. Let the set of same-sign eigenvectors be represented by $\Omega = \{\omega | \lambda_\omega > 0 \wedge [(\mathbf{v}^*_\omega(i) > 0 \forall i) \vee \mathbf{v}^*_\omega(i) < 0 \forall i])\}$. Since the same-sign eigenvectors are orthogonal, this means that there is only one value of ω for which $\mathbf{v}^*_\omega(i) \neq 0$. In other words, each node i is associated with a unique cluster. We denote the set of nodes assigned to the cluster with modal index ω as $S_\omega = \{i | \mathbf{v}^*_\omega(i) \neq 0\}$.

4. Maximum Likelihood Framework

In this paper, we are interested in exploiting the factorization property of Sarkar and Boyer [14] to develop a maximum likelihood method for updating the similarity-weight matrix W. We commence by facto-rising the likelihood-function over the set of modal clusters of the similarity-weight matrix. Since the set of modal clusters are disjoint we can write:

$$P(W) = \prod_{\omega \in \Omega} P(\Phi_\omega), \tag{4}$$

where $P(\Phi_\omega)$ is the probability distribution for the set of similarity-weights belonging to the modal-cluster indexed ω. To model the component probability distributions, we introduce a cluster membership indicator $s_{i\omega}$ which models the degree of affinity of the graph indexed i to the cluster with modal index ω.

Using these variables, we develop a model of probability distribution for the similarity-weights associated with the individual clusters. We assume that the distribution can be factorized over the set of pairwise associations $\Phi_\omega = S_\omega \times S_\omega - \{(i,i) | i \in M\}$ with each cluster and write

$$P(\Phi_\omega) = \prod_{(i,j) \in \Phi_\omega} P(W_{i,j}). \tag{5}$$

To model the probability distribution for the individual link-weights, we adopt the Bernoulli distribution

$$p(W_{i,j}) = W_{i,j}^{s_{i\omega} s_{j\omega}} (1 - W_{i,j})^{1 - s_{i\omega} s_{j\omega}}. \tag{6}$$

This distribution takes on its largest values when either the similarity weight W_{ij} is unity and $s_{i\omega} = s_{j\omega} = 1$, or if the similarity-weight $W_{i,j} = 0$ and $s_{i\omega} = s_{j\omega} = 0$.

With these ingredients the log-likelihood function for the observed pattern of similarity-weights is:

$$\mathcal{L} = \sum_{\omega \in \Omega} \sum_{(i,j) \in \Phi_\omega} \left\{ s_{i\omega} s_{j\omega} \ln W_{ij} + (1 - s_{i\omega} s_{j\omega}) \ln (1 - W_{i,j}) \right\}. \tag{7}$$

Posed in this way the structure of the log-likelihood function has two features which are reminiscent of the expectation-maximization algorithm. First, the modes of the link-weight matrix play the role of mixing components. The product of cluster-membership variables $s_{i\omega} s_{j\omega}$ plays the role of an *a posteriori* measurement probability. Second, the similarity-weights are the parameters which must be estimated. However, there are important differences. The most important of these is that the modal clusters are disjoint. As a result there is no mixing between them.

Based on this observation, we will exploit an EM-like process to update the similarity-weights and the cluster-membership variables. In the "M" step we will locate maximum likelihood similarity-weights. In the "E" step we will use the revised similarity-weight matrix to update the modal clusters. To this end we index the similarity-weights and cluster memberships with iteration number and aim to optimize the quantity

$$Q(W^{(n+1)} | W^{(n)}) = \sum_{\omega \in \Omega} \sum_{(i,j) \in \Phi_\omega} \left\{ s^{(n)}_{i\omega} s^{(n)}_{j\omega} \ln \frac{W^{(n+1)}_{ij}}{1 - W^{(n+1)}_{ij}} + \ln (1 - W^{(n+1)}_{i,j}) \right\}. \tag{8}$$

The revised similarity-weights are indexed at iteration $n+1$ while the cluster-memberships are indexed at iteration n.

4.1. Expectation

To update the cluster-membership variables we have used a gradient-based method. We have computed the derivatives of the expected log-likelihood function with respect to the cluster-membership variable

$$\frac{\partial Q(W^{(n+1)} | W^{(n)})}{\partial s^{(n+1)}_{i\omega}} = \sum_{j \in S_\omega} s^{(n)}_{j\omega} \ln \frac{W^{(n+1)}_{ij}}{1 - W^{(n+1)}_{ij}}. \tag{9}$$

Since the associated saddle-point equations are not tractable in closed form, we use the soft-assign ansatz to update the cluster membership assignment variables. As a result the update equation for the cluster membership indicator variables is:

$$s_{i\omega}^{(n+1)} = \frac{\prod_{j\in S_\omega}\left\{\frac{W_{i,j}^{(n+1)}}{1-W_{ij}^{(n+1)}}\right\}^{s_{j\omega}^{(n)}}}{\sum_{i\in S_\omega}\prod_{j\in S_\omega}\left\{\frac{W_{ij}^{(n+1)}}{1-W_{ij}^{(n+1)}}\right\}^{s_{j\omega}^{(n)}}}, \quad (10)$$

we initialize the cluster membership variables using the same sign eigenvectors and set

$$s_{iw_0} = \frac{|\mathbf{v}_{\omega_0}^*(i)|}{\sum_{i\in S_{\omega_0}}|\mathbf{v}_{\omega_0}^*(i)|}. \quad (11)$$

4.2. Maximization

Once the revised cluster membership variables are to hand then we can apply the maximization step of the algorithm to update the similarity-weight matrix. The updated similarity-weights are found by computing the derivatives of the expected log-likelihood function

$$\frac{\partial Q(W^{(n+1)}|W^{(n)})}{\partial W_{ij}^{(n+1)}} = \sum_{\omega\in\Omega}\left\{s_{i\omega}^{(n)}s_{j\omega}^{(n)}\frac{1}{W_{ij}^{(n+1)}(1-W_{ij}^{(n+1)})} - \frac{1}{1-W_{ij}^{(n+1)}}\right\} \quad (12)$$

and solving the saddle-point equations $\frac{\partial Q(W^{(n+1)}|W^{(n)})}{\partial W_{ij}^{(n+1)}} = 0$. As a result the updated link-weights are given by $W_{ij}^{(n+1)} = \frac{1}{|\Omega|}\sum_{\omega\in\Omega}s_{i\omega}^{(n)}s_{j\omega}^{(n)}$. In other words, the similarity-weight for the pair of nodes (i,j) is simply the average of the product of individual node cluster memberships. Since each graph is associated with a unique cluster, this means that the updated similarity-weight matrix is composed of non-overlapping blocks. Moreover, the similarity-weights are are guaranteed to be in the interval $[0,1]$.

4.3. Algorithm description

Finally, to summarize, the iterative steps of the algorithm are as follows:

- (1): *Initialization*: Compute the eigenvectors of the initial current link-weight matrix $W^{(0)}$. Each same-sign eigenvector whose eigenvalue is positive is used to seed a different component of the mixture model.
- (2) *Expectation*: Compute the updated cluster-membership variables using the E-step.
- (3) *Maximization*: Update the link-weights using the M-step to compute the updated link weight matrix $W^{(n)}$.
- Repeat steps (2) and (3) until convergence is reached.

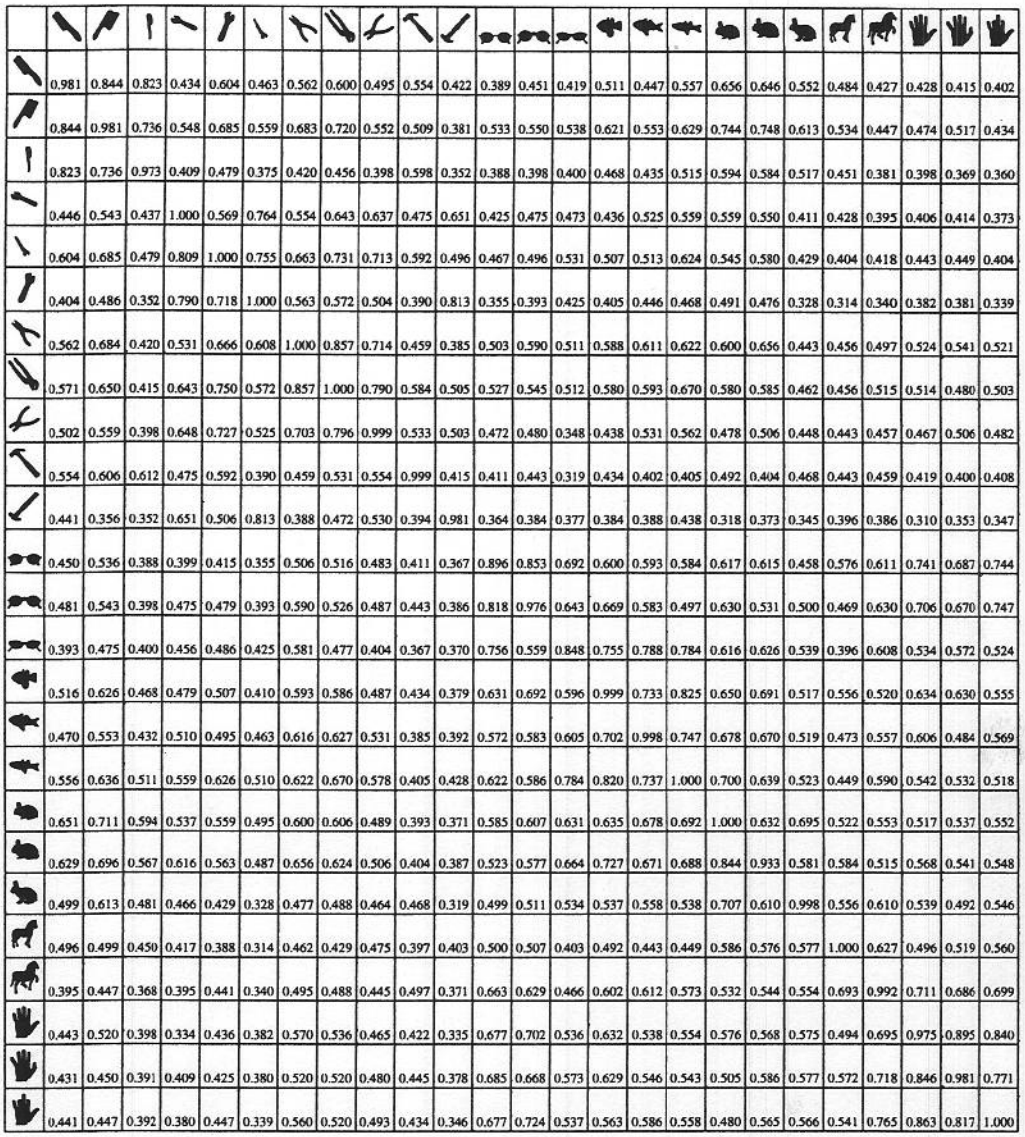
5 Graph-distances

In this section, we provide details of how to compute the distances required in the two graph-clustering applications explored in the experimental section of this paper.

5.1. Shock Tree Edit Distance

The first practical problem tackled in this paper is the clustering of 2D binary shapes based on the similarity of their shock-trees. The idea of characterizing boundary shape using the differential singularities of the reaction equation was first introduced into the computer vision literature by Kimia, Tannenbaum and Zucker [15]. The idea is to evolve the boundary of an object to a canonical skeletal form using the reaction-diffusion equation. The skeleton represents the singularities in the curve evolution, where inward moving boundaries collide. The reaction component of the boundary motion corresponds to morphological erosion of the boundary, while the diffusion component introduces curvature dependent boundary smoothing. Once the skeleton is to hand, the next step is to devise ways of using it to characterize the shape of the original boundary. Here we follow Zucker, Siddiqi, and others, by labeling points on the skeleton using so-called shock-classes [16]. According to this taxonomy of local differential structure, there are different classes associated with behavior of the radius of the maximal circle conatined within the shape. The so-called shocks distinguish between the cases where the local maximal circle has maximum radius, minimum radius, constant radius or a radius which is strictly increasing or decreasing. We abstract the skeletons as trees in which the level in the tree is determined by their time of formation [17, 16]. The later the time of formation, and hence their proximity to the center of the shape, the higher the shock in the hierarchy. While this temporal notion of relevance can work well with isolated shocks (maxima and minima of the radius function), it fails on monotonically increasing or decreasing shock groups. To give an example, a protrusion that ends on a vertex will always have the earliest time of creation, regardless of its relative relevance to the shape.

To overcome this drawback, we augment the structural information given by the skeleton topology and the relative time of shock formation, with a measure of feature importance. We opt to use a shape-measure based on the rate of change of boundary length with distance along the skeleton. To compute the measure we construct the maximal circle,



	0.981	0.844	0.823	0.434	0.604	0.463	0.562	0.600	0.495	0.554	0.422	0.389	0.451	0.419	0.511	0.447	0.557	0.656	0.646	0.552	0.484	0.427	0.428	0.415	0.402
	0.844	0.981	0.736	0.548	0.685	0.559	0.683	0.720	0.552	0.509	0.381	0.533	0.550	0.538	0.621	0.553	0.629	0.744	0.748	0.613	0.534	0.447	0.474	0.517	0.434
	0.823	0.736	0.973	0.409	0.479	0.375	0.420	0.456	0.398	0.598	0.352	0.388	0.398	0.400	0.468	0.435	0.515	0.594	0.584	0.517	0.451	0.381	0.398	0.369	0.360
	0.446	0.543	0.437	1.000	0.569	0.764	0.554	0.643	0.637	0.475	0.651	0.425	0.475	0.473	0.436	0.525	0.559	0.559	0.550	0.411	0.428	0.395	0.406	0.414	0.373
	0.604	0.685	0.479	0.809	1.000	0.755	0.663	0.731	0.713	0.592	0.496	0.467	0.496	0.531	0.507	0.513	0.624	0.545	0.580	0.429	0.404	0.418	0.443	0.449	0.404
	0.404	0.486	0.352	0.790	0.718	1.000	0.563	0.572	0.504	0.390	0.813	0.355	0.393	0.425	0.405	0.446	0.468	0.491	0.476	0.328	0.314	0.340	0.382	0.381	0.339
	0.562	0.684	0.420	0.531	0.666	0.608	1.000	0.857	0.714	0.459	0.385	0.503	0.590	0.511	0.588	0.611	0.622	0.600	0.656	0.443	0.456	0.497	0.524	0.541	0.521
	0.571	0.650	0.415	0.643	0.750	0.572	0.857	1.000	0.790	0.584	0.505	0.527	0.545	0.512	0.580	0.593	0.670	0.580	0.585	0.462	0.456	0.515	0.514	0.480	0.503
	0.502	0.559	0.398	0.648	0.727	0.525	0.703	0.796	0.999	0.533	0.503	0.472	0.480	0.348	0.438	0.531	0.562	0.478	0.506	0.448	0.443	0.457	0.467	0.506	0.482
	0.554	0.606	0.612	0.475	0.592	0.390	0.459	0.531	0.554	0.999	0.415	0.411	0.443	0.319	0.434	0.402	0.405	0.492	0.404	0.468	0.443	0.459	0.419	0.400	0.408
	0.441	0.356	0.352	0.651	0.506	0.813	0.388	0.472	0.530	0.394	0.981	0.364	0.384	0.377	0.384	0.388	0.438	0.318	0.373	0.345	0.396	0.386	0.310	0.353	0.347
	0.450	0.536	0.388	0.399	0.415	0.355	0.506	0.516	0.483	0.411	0.367	0.896	0.853	0.692	0.600	0.593	0.584	0.617	0.615	0.458	0.576	0.611	0.741	0.687	0.744
	0.481	0.543	0.398	0.475	0.479	0.393	0.590	0.526	0.487	0.443	0.386	0.818	0.976	0.643	0.669	0.583	0.497	0.630	0.531	0.500	0.469	0.630	0.706	0.670	0.747
	0.393	0.475	0.400	0.456	0.486	0.425	0.581	0.477	0.404	0.367	0.370	0.756	0.559	0.848	0.755	0.788	0.784	0.616	0.626	0.539	0.396	0.608	0.534	0.572	0.524
	0.516	0.626	0.468	0.479	0.507	0.410	0.593	0.586	0.487	0.434	0.379	0.631	0.692	0.596	0.999	0.733	0.825	0.650	0.691	0.517	0.556	0.520	0.634	0.630	0.555
	0.470	0.553	0.432	0.510	0.495	0.463	0.616	0.627	0.531	0.385	0.392	0.572	0.583	0.605	0.702	0.998	0.747	0.678	0.670	0.519	0.473	0.557	0.606	0.484	0.569
	0.556	0.636	0.511	0.559	0.626	0.510	0.622	0.670	0.578	0.405	0.428	0.622	0.586	0.784	0.820	0.737	1.000	0.700	0.639	0.523	0.449	0.590	0.542	0.532	0.518
	0.651	0.711	0.594	0.537	0.559	0.495	0.600	0.606	0.489	0.393	0.371	0.585	0.607	0.631	0.635	0.678	0.692	1.000	0.632	0.695	0.522	0.553	0.517	0.537	0.552
	0.629	0.696	0.567	0.616	0.563	0.487	0.656	0.624	0.506	0.404	0.387	0.523	0.577	0.664	0.727	0.671	0.688	0.844	0.933	0.581	0.584	0.515	0.568	0.541	0.548
	0.499	0.613	0.481	0.466	0.429	0.328	0.477	0.488	0.464	0.468	0.319	0.499	0.511	0.534	0.537	0.558	0.538	0.707	0.610	0.998	0.556	0.610	0.539	0.492	0.546
	0.496	0.499	0.450	0.417	0.388	0.314	0.462	0.429	0.475	0.397	0.403	0.500	0.507	0.403	0.492	0.443	0.449	0.586	0.576	0.577	1.000	0.627	0.496	0.519	0.560
	0.395	0.447	0.368	0.395	0.441	0.340	0.495	0.488	0.445	0.497	0.371	0.663	0.629	0.466	0.602	0.612	0.573	0.532	0.544	0.554	0.693	0.992	0.711	0.686	0.699
	0.443	0.520	0.398	0.334	0.436	0.382	0.570	0.536	0.465	0.422	0.335	0.677	0.702	0.536	0.632	0.538	0.554	0.576	0.568	0.575	0.494	0.695	0.975	0.895	0.840
	0.431	0.450	0.391	0.409	0.425	0.380	0.520	0.520	0.480	0.445	0.378	0.685	0.668	0.573	0.629	0.546	0.543	0.505	0.586	0.577	0.572	0.718	0.846	0.981	0.771
	0.441	0.447	0.392	0.380	0.447	0.339	0.560	0.520	0.493	0.434	0.346	0.677	0.724	0.537	0.563	0.586	0.558	0.480	0.565	0.566	0.541	0.765	0.863	0.817	1.000

Figure 1. Pairwise affinities computed using weighted tree edit distance.

which is tangent to the two nearest boundary points at each location on the skeleton.

This measurement has previously been used in the literature to express *relevance* of a branch when extracting or pruning the skeleton, but is has recently been shown that its geometric and differential properties make it a good measure of shape similarity [19].

Given this representation we can cast the problem of computing distances between different shapes as that of finding the tree edit distance between the weighted graphs for their skeletons.

Tree edit distance is a generalization to trees of *String edit distance*. The edit distance is based on the existence of a set B of basic edit operation on a tree and a set C of costs, where $c_b \in C$ is the cost of performing the edit operation $b \in B$. The choice of the basic edit operations, as well as their cost, can be tailored to the problem, but common operations include leaf pruning, path merging, and, in case of an attributed tree, change of attribute. Given two trees T_1 and T_2, the set B of basic edit operations, and the cost of such operation $C = c_b, b \in N$, we call an *edit path* from T_1 to T_2 a sequence $b_1, \ldots, b_n$ of basic edit operations that transform T_1 into T_2. The length of such path is $l = c_{b_1} + \cdots + c_{b_n}$; the *minimum length edit path* from T_1 to T_2 is the path form T_1 to T_2 with minimum length. The length of the minimum length path is the tree edit distance.

With our measure assigned to each edge of the tree, we define the cost of matching two edges as the difference of

the total length ratio measure along the branches. The cost of eliminating an edge is equivalent to the cost of matching it to an edge with zero weight, i.e. one along which the total length ratio is zero.

Using the edit distance of the shock trees we generate a similarity measure by weighting the nodes with the border length ratio normalized by the total length of the border of the shape. That is the length of the fraction of the border spanned by the shock group divided by the total length of the border. In this way the sum of the weights in a tree is 1 and the measure is scale invariant. The similarity of the shapes is computed by adding the minimum weight for each matched node, that is $d_{w,w'} = \sum_i min(w_i, w_i')$, where w_i and w_i' are the weight of the nodes that are matched together by our tree edit distance algorithm.

5.2. View Graphs

The second practical problem studied is that of locating the view-structure of 3D objects from 2D images captured from different poses. The image sequence used in our study is taken from the CMU/VASC database and consists of a series of views of a model-house. The images are shown in Figure 3. The graphs used in this part of our study are constructed by extracting corner features and computing their Delaunay triangulations. The extracted graphs are shown in Figure 4. We use a simple matrix method to compute the distance between pairs of graphs corresponding to different viewpoints [21].

To be more formal, we are interested in learning the view structure from a set of images $I_1, I_2, .., I_i, ..., I_N$ whose point-features have been abstracted using Delaunay graphs. Suppose that the features in the image I_i have been abstracted using the graph $G_i = (V_i, E_i)$. Here V_i is denotes the of nodes, i.e. the index-set for the point-features and $E_i \subseteq V_i \times V_i$ is the edge-set for the Delaunay graph. The graph index $i = 1, ...N$ runs over the set of images in their view-order. For each graph G_i, we construct the adjacency matrix A_i. This is a $|V_i| \times |V_i|$ matrix whose element with row index a and column index b is:

$$A_i(a,b) = \begin{cases} 1 & \text{if } (a,b) \in E_i \\ 0 & \text{otherwise.} \end{cases} \quad (13)$$

For the graphs G_i and G_j the distance is:

$$d_{i,j} = Tr[A_i^T (I - C) A_j C^T], \quad (14)$$

where C is a $|V_j| \times |V_j|$ matrix of correspondence indicators and I is a matrix whose elements are each unity. The matrix of correspondence indicators C is found by using matrix factorization to minimize $d_{i,j}$ [21].

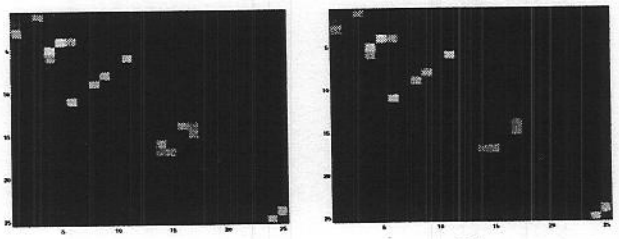
6. Experiments

In this section we experiment with the application of our clustering algorithm to shock graphs and Delaunay triangulations.

6.1. Shock Graphs

The silhouettes used to generate the shock graphs used in our experiments are shown in Figure 1. There are 25 different shapes. These include brushes, tools, spectacles, various animals and human hands. The table lists the initial values of the affinity weight matrix W^0.

A 2D visualization of the set of tree edit-distances for the shapes obtained using multidimensional scaling is shown in figure 2. This illustrates the difficulty of the clustering problem, since no clear structure emerges.



(a) (b)

Figure 3. (a) Initial similarity matrix for the weighted tree edit distances; (b) Final similarity matrix for the weighted tree edit distances.

Figure 3a shows the matrix of pairwise similarity weights for the weighted trees for the different shapes. Here the redder the entry, the stronger the similarity; the bluer the entry, the weaker the similarity. The order of the entries in the matrix is the same as the order of the shapes in Figures 1. After seven iterations of the clustering algorithm the similarity weight matrix shown in Figure 3b is obtained. Distinct clusters appear as the seven extracted clusters, in order of extraction, are:

- 
- 
- 
-
-
- 
-

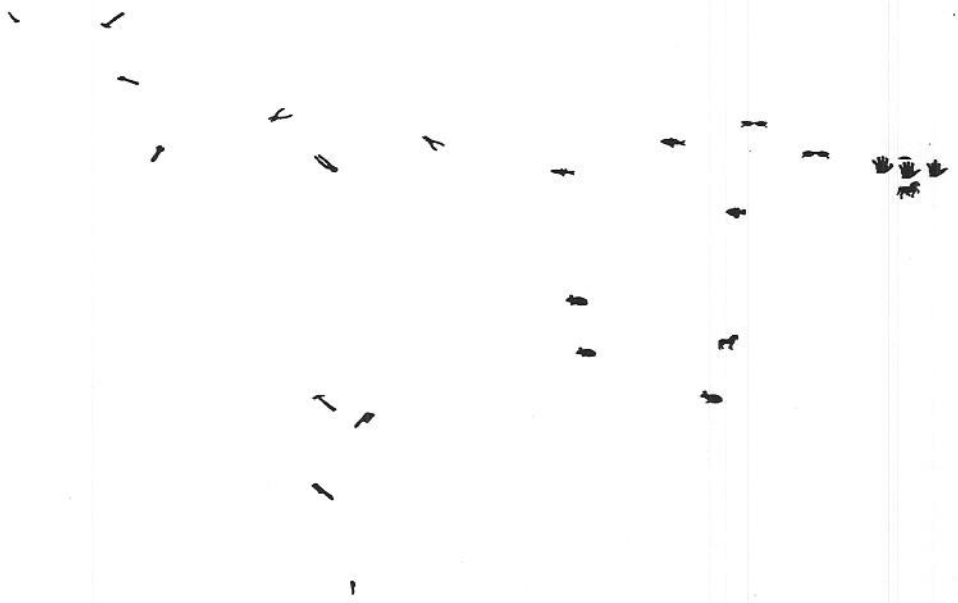


Figure 2. First and second principal component of the edit distances of the shapes.

In other words, the hands, tools, spectacles and animals form clusters. However, there are shapes which leak between these clusters.

The problems encountered above are due to the fact that certain shapes straddle the true shape-classes and cause cluster-merging. When a pruned set of 16 shapes is used, then the following set of clusters emerges:

This is a much better set of clusters that reflect the true shape-classes in the data. This would suggest that more effort needs to be expended in developing a better tree edit-distance for the shock-graphs.

6.2. Delaunay Graphs

In Figure 4 we show the different views of a model house used in our view clustering experiments. The features used to abstract the images as graphs are corners. Overlayed on the images of the Delaunay triangulations of the corners, i.e. the graphs used in our experiments. In Figure 5 we show the initial and final pairwise similarity matrices for the graphs. Here we locate two clusters. The first of these corresponds to the first three graphs. The second corresponds to the remaining seven graphs. The cluster boundary corresponds to a transition in which the front face of the house changes its direction of affine skew.

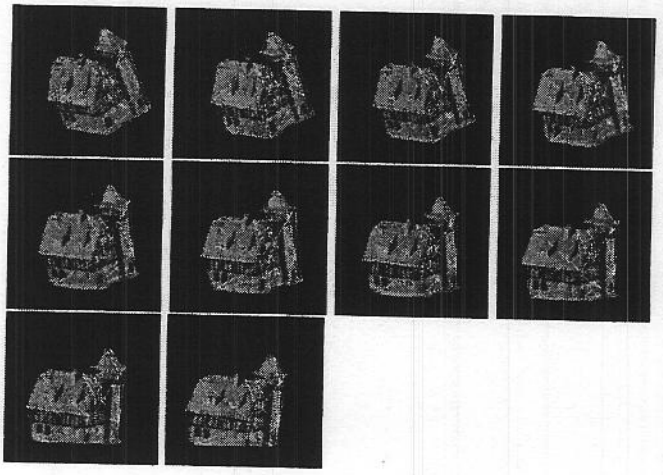
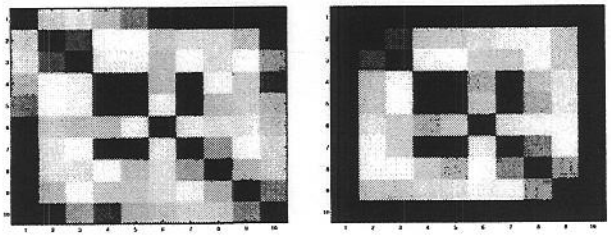


Figure 4. House view graphs.

7. Conclusions

This paper has presented a study of the problem of clustering graphs. We have investigated two problems the first

of these involves shock-trees. Here we gauge the similarity of the trees using weighted edit distance. The second problem involves clustering the Delaunay triangulations of corner features extracted from polyhedral objects in different poses. Here graph-distances are computed using a matrix-based factorization method.



(a) (b)

Figure 5. (a) Initial similarity matrix for the house distances; (b) Final similarity matrix for the house distances.

To identify distinct groups of graphs, we develop a maximum likelihood algorithm for pairwise clustering. This takes as its input, a matrix of pairwise similarities between shock-trees computed from the edit distances. The algorithm is reminiscent of the EM algorithm and has interleaved iterative steps for computing cluster-memberships and for updating the pairwise similarity matrix. The number of clusters is contolled by the number of same-sign eigenvectors of the current similarity matrix. Experimental evaluation of the method shows that it is capable of extracting clusters of trees or graphs which correspond closely to the shape-categories present.

References

[1] S. Rizzi. Genetic operators for hierarchical graph clustering. *Pattern Recognition Letters*, 19:1293–1300, 1998.

[2] J. Segen. Learning graph models of shape. In J. Laird, editor, *Proceedings of the Fifth International Conference on Machine Learning*, pages 29–25, 1988.

[3] R. Englert and R. Glantz. Towards the clustering of graphs. In *2nd IAPR-TC-15 Workshop on Graph-Based Representations*, 1999.

[4] K. Sengupta and K. L. Boyer. Organizing large structural modelbases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4), 1995.

[5] A. Munger H. Bunke and X. Jiang. Combinatorial search vs. genetic algorithms: A case study based on the generalized median graph problem. *Pattern Recognition Letters*, 20(11-13):1271–1279, 1999.

[6] L. G. Shapiro and R. M. Haralick. Relational models for scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4:595–602, 82.

[7] A. Sanfeliu and K. S. Fu. A distance measure between attributed relational graphs fro pattern recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 13:353–362, 1983.

[8] A. K. C. Wong and M. You. Entropy and distance of random graphs with application to structural pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7:599–609, 1985.

[9] K. L. Boyer and A. C. Kak. Structural stereopsis for 3-d vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:144–166, 1988.

[10] J. Kittler W. J. Christmas and M. Petrou. Structural matching in computer vision using probabilistic relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):749–764, 1995.

[11] R. Wilson and E. R. Hancock. Structural matching by discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):634–648, 1997.

[12] K. Sengupta and K. L. Boyer. Modelbase partitioning using property matris spectra. *Computer Vision and Image Understanding*, 70(2), 1998.

[13] T. Hofmann and M. Buhmann. Pairwise data clustering by deterministic annealing. *IEEE Tansactions on Pattern Analysis and Machine Intelligence*, 19(1), 1997.

[14] S. Sarkar and K. L. Boyer. Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors. *Computer Vision and Image Understanding*, 71(1):110–136, 1998.

[15] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker. Shapes, shocks, and deforamtions i. *International Journal of Computer Vision*, 15:189–224, 1995.

[16] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 35(1):13–32, 1999.

[17] A. Shokoufandeh et al. Indexing using a spectral encoding of topological structure. In *Conference on Computer Vision and Pattern Recognition*, June 1999.

[18] A. Robles-Kelly and E. R. Hancock. A maximum likelihood framework for iterative eigendecomposition. In *International Conference on Computer Vision*, Vol. I, pages 654–661, 2001.

[19] A. Torsello and E.R. Hancock. A skeletal measure of 2D shape similarity. In *Visual Form 2001*, LNCS 2059, pages 260–271, 2001.

[20] B. Luo, A.D.J. Cross, and E.R. Hancock. Corner detection via topographic analysis of vector potential. *Pattern Recognition Letters*, 20:635–650, 1999.

[21] B. Luo and E.R. Hancock. Structural graph matching using the EM algorithm and Singular Value Decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:1120–1137, October 2001.