Clustering Shock Trees

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.
erh@cs.york.ac.uk

Abstract
This paper presents a new algorithm for graph-clustering. We adopt maximum likelihood framework and develop an EM-like algorithm. In the E-step cluster memberships are computed using a Bernoulli distribution for the pattern of pairwise similarities between graphs. The M-step updates the matrix of pairwise similarities between graphs. The eigen-modes of the similarity matrix are used to control the number of active clusters. The method is demonstrated on the problem of locating shape-categories in a data-base of shock-graphs.

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 the 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.

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

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 [8] 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 [9] have computed the entropy for random graphs, while Boyer and Kak [10] have used mutual information. More recently, Christmas, Kittler and Petrou [11], and Wilson and Hancock [12] have developed probabilistic measures of graph-similarity. Turning our attention to pairwise clustering, the 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 [14] to induce a classification tree on the data. Finally, there are mean-field methods which can be used to iteratively compute cluster-membership weights [15]. These methods require that the number of pairwise clusters be known a priori.

In this paper we focus on an application involving 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 and un-weighted tree edit distance. We make two theoretical contributions. First, we present a way to compute tree-edit distance. Second, we describe a new maximum-likelihood framework for pairwise clustering in which the number of clusters is controlled using the eigen-modes of the matrix of pairwise similarities. Our experiments show that the best set of shape categories are located when we use weighted edit distance.

2. Shock Tree Edit Distance

The 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 [16]. 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 [18]. According to this taxonomy of local differential structure, there are different classes associated with behavior of the radius of the osculating circle from the skeleton to the nearest pair of boundary points. The so-called shocks distinguish between the cases where the local osculating 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 [19, 18]. 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 osculating circle 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 [20].

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 $l = c_{b_1} + \ldots + c_{b_n}$; the

	0.981	0.844	0.434	0.604	0.562	0.600	0.497	0.554	0.422	0.511	0.557	0.484	0.427	0.428	0.415	0.402
	0.844	0.981	0.548	0.685	0.683	0.720	0.559	0.509	0.381	0.621	0.629	0.534	0.447	0.474	0.517	0.434
	0.446	0.543	1.000	0.569	0.554	0.643	0.637	0.475	0.651	0.436	0.559	0.428	0.395	0.406	0.414	0.373
	0.604	0.685	0.809	1.000	0.663	0.731	0.713	0.592	0.496	0.507	0.624	0.404	0.418	0.443	0.449	0.404
	0.562	0.684	0.531	0.666	1.000	0.857	0.714	0.459	0.385	0.588	0.622	0.456	0.497	0.524	0.541	0.521
	0.571	0.650	0.643	0.750	0.857	1.000	0.793	0.584	0.505	0.580	0.670	0.456	0.515	0.514	0.480	0.503
	0.502	0.559	0.648	0.727	0.703	0.796	1.000	0.533	0.503	0.438	0.562	0.433	0.457	0.467	0.506	0.482
	0.554	0.606	0.475	0.592	0.459	0.531	0.554	1.000	0.415	0.434	0.405	0.443	0.459	0.419	0.400	0.408
	0.441	0.356	0.676	0.506	0.388	0.472	0.530	0.394	0.981	0.384	0.438	0.396	0.386	0.310	0.353	0.310
	0.516	0.626	0.479	0.507	0.593	0.586	0.487	0.434	0.379	1.000	0.825	0.556	0.520	0.634	0.630	0.555
	0.556	0.636	0.559	0.626	0.622	0.670	0.578	0.405	0.428	0.820	1.000	0.449	0.590	0.542	0.532	0.518
	0.496	0.449	0.417	0.388	0.482	0.429	0.475	0.397	0.403	0.492	0.449	1.000	0.627	0.496	0.519	0.560
	0.395	0.447	0.395	0.441	0.495	0.488	0.445	0.497	0.371	0.602	0.573	0.693	0.992	0.711	0.687	0.699
	0.443	0.520	0.334	0.436	0.570	0.536	0.465	0.422	0.335	0.632	0.554	0.494	0.696	0.976	0.895	0.840
	0.431	0.450	0.409	0.425	0.520	0.520	0.480	0.445	0.378	0.629	0.543	0.572	0.719	0.847	0.982	0.771
	0.441	0.447	0.380	0.447	0.560	0.520	0.493	0.434	0.346	0.563	0.558	0.541	0.765	0.864	0.817	1.000

Fig. 1. Pairwise edit distances computed using un-weighted trees.

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 two similarity measures for a pair of shapes.

- The first measure is obtained 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.

	1.000	1.000	0.774	1.000	0.889	0.889	0.706	0.643	0.850	0.818	0.889	0.635	0.640	0.706	0.694	0.684
	1.000	1.000	0.774	1.000	0.889	0.889	0.706	0.643	0.850	0.818	0.889	0.635	0.640	0.706	0.694	0.684
	0.774	0.774	1.000	0.774	0.833	0.833	0.676	0.714	0.800	0.773	0.694	0.615	0.620	0.676	0.667	0.658
	1.000	1.000	0.774	1.000	0.889	0.889	0.706	0.643	0.850	0.818	0.889	0.635	0.640	0.706	0.694	0.684
	0.889	0.889	0.694	0.889	1.000	1.000	0.765	0.730	0.950	0.808	0.778	0.673	0.680	0.765	0.750	0.737
	0.889	0.889	0.833	0.889	1.000	1.000	0.765	0.730	0.950	0.808	0.778	0.673	0.680	0.765	0.750	0.737
	0.706	0.706	0.676	0.706	0.765	0.765	1.000	0.782	0.794	0.674	0.680	0.730	0.692	0.765	0.801	0.669
	0.643	0.643	0.714	0.643	0.730	0.730	0.847	1.000	0.771	0.649	0.639	0.769	0.724	0.651	0.698	0.682
	0.850	0.850	0.800	0.850	0.950	0.950	0.794	0.857	1.000	0.764	0.739	0.692	0.700	0.794	0.778	0.763
	0.818	0.818	0.773	0.818	0.808	0.808	0.599	0.731	0.764	1.000	0.909	0.647	0.720	0.749	0.806	0.789
	0.889	0.889	0.694	0.889	0.778	0.778	0.680	0.548	0.739	0.909	0.778	0.598	0.680	0.595	0.667	0.737
	0.635	0.635	0.615	0.635	0.673	0.673	0.778	0.769	0.692	0.647	0.598	1.000	0.785	0.730	0.752	0.729
	0.640	0.640	0.620	0.640	0.680	0.680	0.741	0.724	0.700	0.655	0.680	0.824	1.000	0.840	0.764	0.834
	0.706	0.706	0.676	0.706	0.765	0.765	0.706	0.651	0.715	0.824	0.765	0.730	0.840	1.000	0.972	0.947
	0.694	0.694	0.667	0.694	0.750	0.750	0.801	0.698	0.778	0.659	0.583	0.752	0.812	0.915	1.000	0.920
	0.684	0.684	0.658	0.684	0.737	0.737	0.669	0.682	0.763	0.718	0.573	0.729	0.741	0.947	0.920	0.947

Fig. 2. Pairwise edit distances computed using weighted trees.

- The second measure of shape similarity is computed from the unweighted structure: We assign a uniform edit cost of 1 to each node and we compute the average ratio of matched nodes: $d_{w,w'} = \frac{1}{2}\left(\frac{\#\hat{T}}{\#T_1} + \frac{\#\hat{T}}{\#T_2}\right)$, where T_1 and T_2 are the two trees to be matched, $\hat{T}$ is the median of the two trees obtained through cut operations only, and # indicates the number of nodes in the tree.

3. Graph-clustering by Matrix Factorisation

We pose the problem of graph-clustering as that of finding pairwise clusters in the distribution of tree-edit distance. The process of pairwise clustering is somewhat different to the more familiar one of central clustering. Whereas central clustering aims to characterise 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 [15].

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 characterised using a matrix of pairwise similarity weights. The elements of this weight matrix are computed using tree-edit distance $d_{i,j}$ between the graphs indexed i and j. Here we use the exponential similarity function

$$W_{i,j} = \begin{cases} \exp[-kd_{i,j}] & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

to generate the elements of the weight-matrix, where k is a constant which is heuristically set. The aim in graph-clustering is to locate the updated 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 factorisation methods to locate the clusters. One way of viewing this is as the 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 [21] 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 $\underline{x}'W\underline{x}$, where W is the weighted adjacency matrix, is maximised when $\underline{x}$ 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 $\underline{x}_{\lambda_1}, \underline{x}_{\lambda_2}, ...$ are found by solving the equation $W\underline{x}_{\lambda_i} = \lambda_i \underline{x}_{\lambda_i}$. Let the set of same-sign eigenvectors be represented by $\Omega = \{\omega | \lambda_\omega > 0 \wedge [(\underline{x}^*_\omega(i) > 0 \forall i) \vee \underline{x}^*_\omega(i) < 0 \forall i])\}$. Since the same-sign eigenvectors are orthogonal, this means that there is only one value of ω for which $\underline{x}^*_\omega(i) \neq 0$. 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 $S_\omega = \{i | \underline{x}^*_\omega(i) \neq 0\}$.

4. Maximum Likelihood Framework

In this paper, we are interested in exploiting the factorisation property of Sarkar and Boyer [21] to develop a maximum likelihood method for updating

the similarity-weight matrix W. We commence by factorising 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,

$$P(W) = \prod_{\omega \in \Omega} P(\Phi_\omega) \tag{2}$$

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 which models the degree of affinity of the graph indexed i to the cluster with modal index ω. This is done using the magnitudes of the modal co-efficients and we set

$$s_{i\omega} = \frac{|\mathbb{x}^*_\omega(i)|}{\sum_{i \in S_\omega} |\mathbb{x}^*_\omega(i)|} \tag{3}$$

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 factorised over the set of pairwise associations $\Phi_\omega = S_\omega \times S_\omega - \{(i,i)|i \in M\}$ with each cluster and write

$$P(\Phi_\omega) = \prod_{(i,j) \in \Phi_\omega} P(W_{i,j}) \tag{4}$$

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

$$p(W_{i,j}) = W_{i,j}^{s_{i\omega} s_{j\omega}} (1 - W_{i,j})^{1 - s_{i\omega} s_{j\omega}} \tag{5}$$

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

$$\mathcal{L} = \sum_{\omega \in \Omega} \sum_{(i,j) \in \Phi_\omega} \left\{ s_{i\omega} s_{j\omega} \ln W_{ij} + (1 - s_{i\omega} s_{j\omega}) \ln(1 - W_{i,j}) \right\} \tag{6}$$

Posed in this way the structure of the log-likelihood function has two features which are reminiscent of thet expectation-maximisation 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 posteri-ori* 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 optimise the quantity

$$Q(W^{(n+1)}|W^{(n)}) = \sum_{\omega\in\Omega}\sum_{(i,j)\in\Phi_\omega}\left\{s_{i\omega}^{(n)}s_{j\omega}^{(n)}\ln\frac{W_{ij}^{(n+1)}}{1-W_{ij}^{(n+1)}}+\ln(1-W_{i,j}^{(n+1)})\right\} \quad (7)$$

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

$$\frac{\partial Q(W^{(n+1)}|W^{(n)})}{\partial s_{i\omega}^{(n+1)}} = \sum_{j\in S_\omega} s_{j\omega}^{(n)}\ln\frac{W_{ij}^{(n+1)}}{1-W_{ij}^{(n+1)}} \quad (8)$$

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

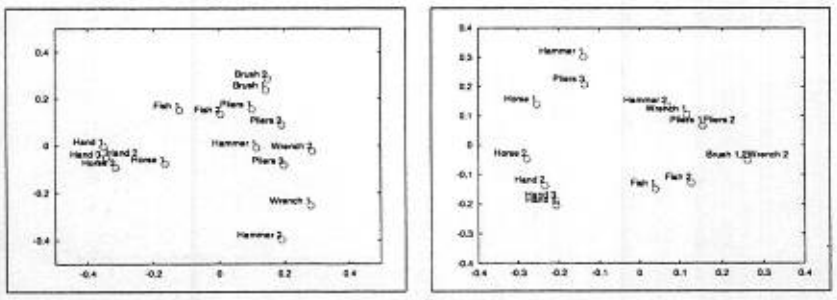
$$s_{i\omega}^{(n+1)} = \frac{\prod_{j\in S_\omega}\left\{\frac{W_{i,j}^{(n+1)}}{1-W_{ij}^{(n+1)}}\right\}^{s_{j\omega}^{(n)}}}{\sum_{i\in S_\omega}\prod_{j\in S_\omega}\left\{\frac{W_{ij}^{(n+1)}}{1-W_{ij}^{(n+1)}}\right\}^{s_{j\omega}^{(n)}}} \quad (9)$$

4.2. Maximisation

Once the revised cluster membership variables are to hand then we can apply the maximisation 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

$$\frac{\partial Q(W^{(n+1)}|W^{(n)})}{\partial W_{ij}^{(n+1)}} = \sum_{\omega \in \Omega} \left\{ s_{i\omega}^{(n)} s_{j\omega}^{(n)} \frac{1}{W_{ij}^{(n+1)}(1 - W_{ij}^{(n+1)})} - \frac{1}{1 - W_{ij}^{(n+1)}} \right\} \tag{10}$$

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 $W_{ij}^{(n+1)} = \sum_{\omega \in \Omega} s_{i\omega}^{(n)} s_{j\omega}^{(n)}$. 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 $[0, 1]$.

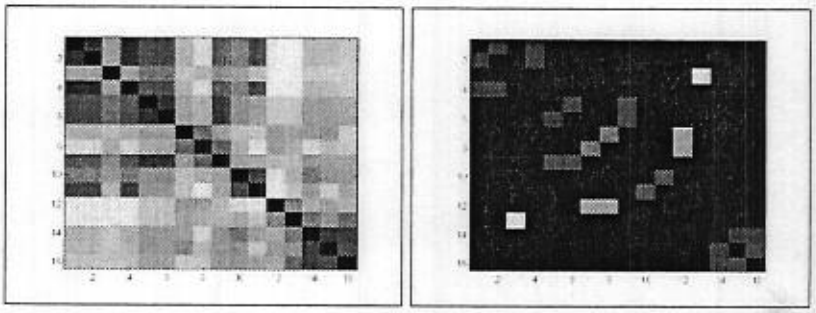


(a) (b)

Fig. 3. (a)Multidimensional scaling applied to the weighted tree edit distances; (b)Multidimensional scaling applied to the unweighted tree edit distances.

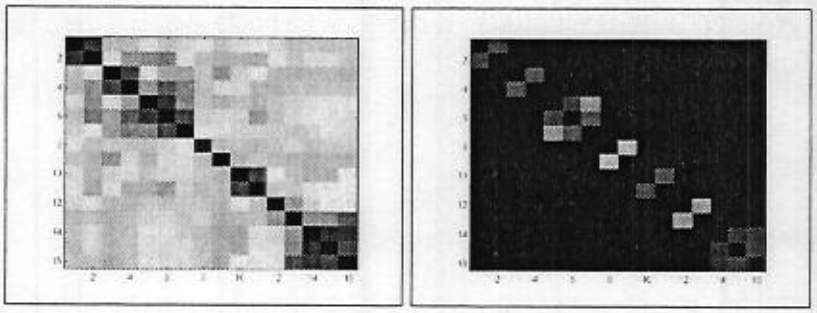
5. Experiments

The 16 shapes used in our study are shown in Figures 1a and 1b. In Figure 1a we show the pattern of weighted edit distances between the shock-trees for the shapes, while Figure 1b shows the corresponding weighted tree edit distances. To visualise the data, we have applied multi-dimensional scaling to the two sets of distances [23]. Figures 2a and 2b respectively show the leading two components of the eigenvectors of the distance matrix for the weighted and unweighted trees. In the case of the weighted shock trees, the cluster structure



(a) (b)

Fig. 4. (a) Initial similarity matrix for the un-weighted tree edit distances; (b)Final similarity matrix for the unweighted tree edit distances.



(a) (b)

Fig. 5. (a) Initial similarity matrix for the weighted tree edit distances; (b) Final similarity matrix for the weighted tree edit distances.

is somewhat clearer than for the un-weighted shock trees. However, in both cases the cluster structure is far from clear. In other words, the clustering of the graphs is not a problem that could be solved by simply applying central clustering to the eigenvectors delivered by MDS.

In Figures 3a we show the matrix of pairwise similarity weights for the un-weighted trees for the different shapes. Here the redder the entries, the stronger the similarity; the bluer the entries, the weaker the similarity. The order of the

entries in the matrix is the same as the order of the shapes in Figures 1a and 1b. After six iterations of the clustering algorithm the similarity weight matrix shown in Figure 3b is obtained. There are six clusters (brush (1) + brush (2) + wrench (4); spanner (3) + horse (13) ; pliers (5) + pliers (6) + hammer (9) ;pliers (7) +hammer (8) + horse (12); fish (10) + fish (12); hand (14) + hand (15) + hand (16). Clearly there is merging and leakage between the different shape categories. In Figures 4a and 4b we show the initial and final similarity matrices when weighted trees are used. The entries in the initial similarity matrix are better grouped than those obtained when the unweighted tree edit distance is used. There are now seven clusters. brush (1) + brush (2) ; spanner (3) + spanner (4); pliers (5) + pliars (6) + pliers (7); hammer (8) + hammer (9); fish (10) + fish (11); horse (12) + horse (13); hand (14) + hand (15) + hand (16)). These correspond exactly to the shape categories in the data-base.

6. Conclusions

This paper has presented a study of the problem of clustering shock-trees. We gauge the similarity of the trees using weighted and unweighted edit distance. To idetify distinct groups of trees, we develop a maixmum 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 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] M. A. Eshera and K. S. Fu. A graph distance measure for image analysis. *IEEE Transactions on Systems, Man and Cybernetics*, 14:398–407, 1984.

[8] 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.

[9] 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.

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

[11] 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.

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

[13] B. Huet and E. R. Hancock. Shape recognition from large image libraries by inexact graph matching. *Pattern Recognition Letters*, 20(11-13):1259–1270, 1999.

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

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

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

[17] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker. The hamilton-jacobi skeleton. In *Seventh International Conference on Computer Vision*, pages 828–834. IEEE, IEEE Computer Society, September 1999.

[18] 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.

[19] A. Shokoufandeh, S. J. Dickinson, K. Siddiqi, and S. W. Zucker. Indexing using a spectral encoding of topological structure. In *Conference on Computer Vision and Pattern Recognition*, June 1999.

[20] A. Torsello and E.R. Hancock. A skeletal measure of 2d shape similarity. Submited, 2000.

[21] 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.

[22] J. S. Bridle. Training stochastic model recognition algorithms can lead to maximum mutual information estimation of parameters. In *NIPS 2*, pages 211–217, 1990.

[23] C. Chatfield and A.J. Colins. *Introduction to multivariate analysis*. Chapman & Hall/CRC, 1980.