Learning Shape-Classes Using a Mixture of Tree-Unions

Andrea Torsello and Edwin R. Hancock

Abstract—This paper poses the problem of tree-clustering as that of fitting a mixture of tree unions to a set of sample trees. The tree-unions are structures from which the individual data samples belonging to a cluster can be obtained by edit operations. The distribution of observed tree nodes in each cluster sample is assumed to be governed by a Bernoulli distribution. The clustering method is designed to operate when the correspondences between nodes are unknown and must be inferred as part of the learning process. We adopt a minimum description length approach to the problem of fitting the mixture model to data. We make maximum-likelihood estimates of the Bernoulli parameters. The tree-unions and the mixing proportions are sought so as to minimize the description length criterion. This is the sum of the negative logarithm of the Bernoulli distribution, and a message-length criterion that encodes both the complexity of the union-trees and the number of mixture components. We locate node correspondences by minimizing the edit distance with the current tree unions, and show that the edit distance is linked to the description length criterion. The method can be applied to both unweighted and weighted trees. We illustrate the utility of the resulting algorithm on the problem of classifying 2D shapes using a shock graph representation.

Index Terms—Structural learning, tree clustering, mixture modelinq, minimum description length, model codes, shock graphs.

1 Introduction

The unsupervised learning of shape from a set of training examples is a problem of considerable topicality and practical importance in the field of computer vision. There are two aspects to the problem. The first of these is that of determining the set of shape-classes present and this is effectively a problem of clustering shapes. The second aspect to the problem is that of characterizing the modes of shape variation present within each class. This latter problem is frequently posed as that of learning a pattern-space for the shapes and has attracted considerable interest in both the computer vision and statistics literature.

These two problems have been extensively studied with geometric characterizations of shape using both simple descriptors such as landmark points on the boundary [4] or more complex ones such as curve descriptors [34]. Shape-classes can be located by vectorizing the shape-attributes and applying standard central clustering techniques to the shape-vectors. The problem of learning the modes of shape-variations present within a class is a more challenging one. A straightforward method is to capture shape-variations by performing principal components analysis on the class covariance matrix [4]. A more sophisticated approach, which can be traced back to the seminal work of Kendall [20], is to construct a shape-manifold [34].

An alternative to the use of boundary or curve attributes is to use a structural abstraction. Here, the idea is to decompose the object under study into component parts and to represent the arrangement of these parts using a relational graph [26], [21], [41]. In particular, trees are frequently used to represent the hierarchical arrangement of the parts of shape-primitives. For instance, trees can be used to represent articulated objects [23], [41], [17]. Recently, there has been interest in the structural representation of 2D and 3D shapes using their medial axes or skeletons. In the shock-tree, nodes represent sections of the skeleton of the shape and edges represent their adjacency relations. The branches of the skeleton are assigned labels that distinguish the shape of the corresponding section of the object boundary [33]. The labels distinguish whether the boundary is of constant width, tapering, locally constricted, or a local bulge. Changes in boundary shape give rise to structural variations in the shock tree.

Unfortunately, the problem of learning shape-classes and shape-variations is much less tractable when a structural abstraction is used. The main obstacle is that graphs are not easily converted to vectors and, hence, standard statistical pattern recognition techniques cannot be easily applied to them. The reason for this is that there is no canonical ordering for the nodes in a graph and a correspondence order must be established before analysis can commence. Moreover, the shape variation present may manifest itself as changes in node and edge structure. Hence, even if a correspondence order can be established, then graphs do not necessarily map to vectors of fixed length.

To overcome these problems, the aim in this paper is to develop an unsupervised method for learning a generative model of tree structure from unlabeled data. We capture class structure using a mixture model over the different shape categories present in the data. Shape variations within each class are captured by using a structure referred to as a union tree. This is a structure from which each tree assigned to a particular class can be obtained using node and, hence, edge, removal operations. The nodes in the structure are assigned weights to indicate their relative importance in the training sample. We show how to fit this mixture model to samples of

- *A. Torsello is with the Dipartimento di Informatica, Universitá Ca' Foscari di Venezia, via Torino 155, 30172 Venezia Mestre, Italy. E-mail: torsello@dsi.unive.it.*
- *E.R. Hancock is with the Department of Computer Science, University of York, Heslington, York, YO10 5DD England. E-mail: erh@cs.york.ac.uk.*

Manuscript received 17 Nov. 2004; revised 14 Oct. 2005; accepted 18 Oct. 2005; published online 13 Apr. 2006.
Recommended for acceptance by L. Kuncheva.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-0614-1104.

tree data by minimizing a description length criterion. In the remainder of this section, we review the literature related to this problem and underline the novelty of our contribution.

1.1 Related Literature

In this paper, we aim to take an information theoretic approach to the problem of estimating the tree model. The literature on description length methods is intricate and the distinction between the different approaches is relatively fine. Broadly speaking, there are two approaches to the problem. The first to be developed was Wallace and Burton's minimum message length (MML) principle [39]. This was followed about a decade later by Risannen's minimum description length formulation (MDL) [29]. The two approaches were jointly presented at a Royal Statistical Society Discussion Meeting in 1987. There is an interesting account of the following discussion, which highlights some of the differences in perspective. More recent and rather different, accounts of the similarities and differences are given in the surveys by Baxter and Oliver [1] and by Lanterman et al. [22]. Both methods revolve around the use of the log-posterior probability to locate a model that is optimal with respect to code length. In MDL, it is the selection of the model that is the focus of attention, and the model parameters are simply an means to this end. In fact, the model parameters are usually taken to be the maximum-likelihood estimates and these may be different from the MDL estimates. In MML, on the other hand, the recovery of the model parameters is a central aim. The two approaches lead to rather different optimization criteria. In MDL, the prior on the model parameters is assumed to be flat, while for MML this is not the case.

The choice of coding scheme is also important. The simplest approach is to assume that the code-length follows an exponential distribution. If this is not the case, then more complex coding schemes including universal coding and predictive coding can be used [13]. Of course, the assumption of an exponential distribution leads to algorithms that are more tractable than universal or predictive codes. In MDL, the codes can be either one part or two part, depending on whether or not the parameters are encoded. In MML, the codes are two-part. MML has been used as a means of fitting mixture models to data [7].

However, it should be stressed that the distinctions between MML and MDL are not clear cut, and our approach is as follows: We aim to fit a mixture of tree union models to a set of sample trees (the data). The individual trees constituting the data are assumed to be sampled from the tree-unions under a Bernoulli model. For each node of each tree union, there is a parameter that describes whether or not the node is observed in a data sample. These parameters are estimated using a standard maximum-likelihood method and this is hence consistent with the MDL approach. Our aim is to find the mixture of tree unions that best accounts for the observed tree sample using a minimum encoding criterion. Each mixture component is a probability distribution over a union tree structure. The tree-union is a structure from which the individual data samples assigned to the corresponding mixture component can be obtained by edit operations. The Bernoulli distribution describes the probability of observing individual nodes in the training samples. The assignment of tree-samples to tree unions is represented by a set of binary indicator variables. The coding criterion corresponding to the mixture model has three parts. The first part describes the tree-data, given the set of tree union parameters together with the cluster membership indicators. This is the negative of the log-probability of the Bernoulli distribution. The second part encodes the membership indicators given the mixture model parameters. This is the negative of the logarithm of the mixing proportions. The final part of the criterion encodes the model parameters. Here, we assume an exponential prior over the size of the model. Our aim is to update both the membership indicators and the tree union so as to optimize this three-part code. Hence, our coding scheme is more reminiscent of those used in MML than those conventionally used in MDL.

The machine learning literature describes a number of attempts to use description length methods to learn graph or tree structure. The main problems addressed are learning Bayesian networks [14], [10], mixtures of tree-classifiers [25], and relational models [11]. In each of these problems, the nodes represent random variables and the edges the dependency between different random variables. In order to estimate the dependency of the random variables and, hence, the structure of the graphical model, Friedman [9] introduces a structural version of the EM algorithm. The method uses a description length criterion to gauge the effects of structural changes in the graph, and model selection is based on the minimum description length (MDL) principle [29]. Building on earlier work on tree-based classifiers by Chow and Liu [3], Meliă [25] develops a model of the distribution of tree structures. Formally, the model is a probabilistic mixture of Bayesian networks, where the topology of each mixture is constrained to be a tree. This restriction allows for efficient inference using a structural variant of the EM algorithm and this has proven to be effective for several classification problems. Here, too, a Bayesian approach is used for model selection. An alternative approach to estimating the underlying representation of structured data is found in the literature on concept learning [8]. Here, the data is composed of identifiable semistructured elements such as words in sentences and the goal is to estimate the underlying relations such as the grammar or the semantic affinity, so as to group observations into concepts that are not known ab initio. A similar approach can be found in [12] where the MDL principle is used to infer a grammar from a sample of sentences.

Although these methods provide a powerful way to infer the relations between the random variables or nodes in the model, they rely on the availability of correspondences between the observations in the training data and the nodes of the structure that is being learned. However, in many situations, the correspondences may not be available. Instead, the correspondences are hidden variables that must be recovered using a graph matching technique during the learning process. Hence, there is a chicken and egg problem in structural learning. Before the structural model can be learned, the correspondences with it must be available and, yet, the model itself must be to hand to locate correspondences. The problem that we wish to address here is complementary to that of learning a graphical model. In the case of a graphical model, the training data is accompanied with complete correspondence information, but the structural information is absent and must be inferred from the data. When learning structural archetypes, on the other hand, the data has structural organization, but correspondence information is lacking and must be estimated using graph matching techniques. Additionally, in the latter problem, the structural information may also be incomplete and noisy.

Recently, however, there has been some effort aimed at learning structural archetypes and clustering data abstracted in terms of graphs. Lozano and Escolano [24], and Bunke et al. [2] summarize the data by creating super-graph representation from the available samples. While these techniques provide a structural model of the samples, the way in which the supergraph is learned or estimated is largely heuristic in nature and draws neither on probabilistic nor information theoretic methods. Jain and Wysotzki, on the other hand, adopt a geometric approach which aims to embed graphs in a high-dimensional space by means of the Schur-Hadamard inner product [19]. Central clustering methods are then deployed to learn the class structure of the graphs. Horváth et al. [16] use edit-distance and pairwise classification to learn logical predicates, while Hagenbuchner et al. [15] use Recursive Neural Networks to perform unsupervised learning of graph structures. While these approaches preserve the structural information present, they do not provide a means of characterizing the modes of structural variation encountered and this renders them of limited use for the analysis of shape.

1.2 Contribution

Hence, our contribution in this paper is to bring together ideas from graph theory and statistical learning theory to develop a method for learning the class prototype and class structure of a set of unlabeled trees. The overall goal is to develop a central clustering method that can be used to assign sample trees to clusters. We apply the resulting framework to the problem of learning a generative model for sets of shock trees. By fitting our mixture of tree-unions to sets of shock trees, we are able to learn the shape classes present in the training data. Moreover, the tree-unions for each class can be used to construct shape-spaces for the shock-trees.

The outline of this paper is as follows: In Section 2, we develop the generative tree model that underpins our graph-clustering method and describe how it may be encoded. Section 3 extends the framework to trees with weights associated to each node. Section 4 turns to details of how the description length criterion may be minimized by refining an initially overspecific model by merging pairs of trees. In Section 5, we explore the relationship between the change in description length gained through tree merge operations and the corresponding tree edit distance. Here, we show that the edit costs are related to the description costs (and, hence, the node probabilities). Section 6 describes how the correspondences can be used to map the trees onto pattern vectors and how PCA may be used to construct pattern spaces for the trees. In Section 7, we provide experiments which demonstrate the utility of our method for the problem of clustering shock trees. Finally, Section 8 offers some conclusions and suggests directions for future work.

2 Generative Tree Model

We commence our development by providing some preliminary definitions of trees and order relations. Let $\mathcal{N}$ be an arbitrary set, a set $\mathcal{O} \subseteq \mathcal{N} \times \mathcal{N}$ is a *strict partial order* over $\mathcal{N}$ if

1. $\forall x \in \mathcal{N} \quad (x,x) \notin \mathcal{O}$ (irreflexivity).
2. $\forall x,y \in \mathcal{N}\ (x,y) \in \mathcal{O} \Rightarrow (y,x) \notin \mathcal{O}$ (antisymmetry).
3. $\forall x,y,z \in \mathcal{N} \quad (x,y) \in \mathcal{O} \wedge (y,z) \in \mathcal{O} \Rightarrow (x,z) \in \mathcal{O}$ (transitivity).

Furthermore, $\mathcal{O}$ is said to be a *tree-order* if

4. $\exists r \in \mathcal{N} \quad \forall x \in \mathcal{N}, x \neq r \quad (r,x) \in \mathcal{O}$.
5. $\forall x,y,z \in \mathcal{N} (x,z) \in \mathcal{O} \wedge (y,z) \in \mathcal{O} \Rightarrow (x,y) \in \mathcal{O} \vee (y,x) \in \mathcal{O}$.

A *hierarchical tree* $T = (\mathcal{N}, \mathcal{O})$ is the tuple consisting of a node-set $\mathcal{N}$ and a tree-order $\mathcal{O}$ over $\mathcal{N}$, which is said to be the *topological order* of T. If $(x,y) \in \mathcal{O}$, then x is said to be an *ancestor* of y and y a *descendant* of x. If $(x,y) \in \mathcal{O} \wedge (\nexists z \in \mathcal{N}\ (x,z) \in \mathcal{O} \wedge (z,y) \in \mathcal{O})$, x is said to be the *parent* of y and y a *child* of x. The node r that satisfies Condition 4 is said to be the *root* of T.

Consider the set of hierarchical trees $\mathcal{D} = \{t_1, t_2, \ldots, t_n\}$. Our aim in this paper is to cluster these trees, i.e., to perform unsupervised learning of the class structure of the sample. We pose this problem as that of learning a mixture of generative class archetypes. Each class archetype is constructed by merging sets of sample trees together to form a superstructure called the *tree-union*. The tree merging process requires node correspondence information and we work under conditions in which these are unknown ab initio and must be inferred as part of the learning process.

2.1 Probabilistic Model

Our aim is to construct a probabilistic model that can be used to describe the distribution of tree data. Formally, the goal is to construct a conditional probability distribution $P(t|\mathcal{H})$ for an observed tree t given the available structural model $\mathcal{H}$. To develop this probabilistic model, we make some simplifying assumptions. First, we assume that the observed tree data can be modeled as a mixture over k tree models $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$, i.e.,

$$P(t|\mathcal{H}) = \sum_{c=1}^{k} \alpha_c P(t|\mathcal{T}_c),$$

where $\{\alpha_1, \ldots, \alpha_k\}$ are the mixing proportions. Second, an observed tree t is assumed to have been generated by the mixture component $\mathcal{T}_c$ through a simple node observation model, where sampling error acts only on nodes, while hierarchical relations are assumed always to be sampled correctly. More formally, the tree model $\mathcal{T}_c$ is specified by a set of nodes $\mathcal{N}_c$, a tree order $\mathcal{O}_c$ and a set of node-observation probabilities $\Theta_c = \{\theta_{c,i}\}_{i \in \mathcal{N}_c}$, where $\theta_{i,c}$ is the probability of observing the node i in the set of trees assigned to the class c. An observation from a model $\mathcal{T}_c$ is a tree t with node-set $\mathcal{N}^t \subset \mathcal{N}_c$ and topological order $\mathcal{O}^t = \mathcal{O}_c \cap (\mathcal{N}^t \times \mathcal{N}^t)$. We assume that the individual nodes of the observed tree t are sampled from the model tree $\mathcal{T}_c$ as independent Bernoulli trials. According to this model, the node i in trees belonging to class c is observed with probability $\theta_{i,c}$ and fails to be observed with probability $1 - \theta_{i,c}$. As a result, the tree-model $\mathcal{T}_c$ generates the observed tree t with probability

$$P(t|\mathcal{T}) = \begin{cases} \prod_{i \in \mathcal{N}^t} \theta_{c,i} \prod_{j \in \mathcal{N}_c \setminus \mathcal{N}^t} (1 - \theta_{c,j}) & \text{if } t \text{ is an observation of } \mathcal{T}_c \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

In other words, the probability distribution is factored into $|\mathcal{N}_c|$ independent node observations. The interpretation of this model is that $\mathcal{T}_c$ represents the true tree structure, but the

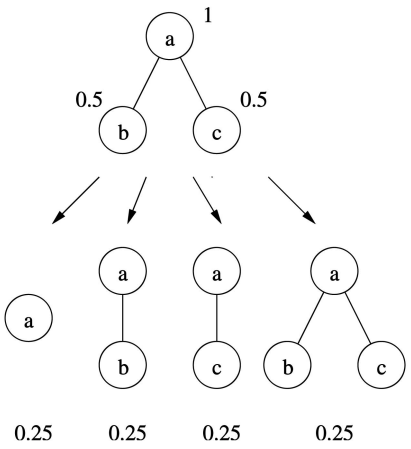

Fig. 1. An example of tree observation. The top structure represents a tree model, the figures besides each node is the node's sampling probability and the labels indicate node identity. The bottom structures represent the trees generated by the model, the figures below each tree is the probability of observing it. Note that the second and third trees are structurally indistinguishable.

observation process is such that noise or sampling errors cause some nodes to go undetected by the observer. The error process is independent over the nodes of the tree. As an example, consider the tree model composed of a root and two children with node sampling probabilities 1 for the parent (root) and 0.5 for each child. This model can produce four different observed trees and each of these occurs with probability 0.25 (see Fig. 1).

Clearly, the structure produced by the observation model is always acyclic. The reason for this that the original model is acyclic and the observation process can only eliminate edges. However, if the root is not observed, then the observed sample might not be connected, and as result is a forest rather than a tree. To avoid this potential problem, we assume that the probability of observing the root is always 1. In this way, the observation model can only eliminate lower level nodes and the tree will always remain connected. In practice, enforcing the observation of the root is not problematic in our application domain. The reason for this is that shock trees require the addition of a dummy root. However, the dummy root is not actually part of the skeleton from which the shock tree is derived (it is simply the barycenter of the shape). As a result, the root node is always present. We work under the assumption that node correspondences, except perhaps the root, are not known ab initio and must be inferred from the structural relations that survive in the observed trees. Returning to the example in Fig. 1, this means that the second and third tree observations are indistinguishable.

The aim of this paper is to estimate the structural model $\mathcal{H}$ and, hence, the tree models $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$, that best fit the set of observed tree data $\mathcal{D}$. The requirement that the correspondences must be inferred from the structural relations in the observed trees, means that the conditional probability model is dependent on the set of correspondences $\mathcal{C}$ between the nodes of the observed tree and the nodes of the model tree. That is, the correspondences provide information concerning the identity of the model-node that generated a particular node in an observed tree. Hence, the selected correspondences influence the estimation of the node sampling probabilities. Clearly, for the tree t to be valid observation of the model $\mathcal{T}$, the correspondences $\mathcal{C}$ must satisfy the hierarchical constraints prevailing in $\mathcal{T}$, that is

$$\begin{aligned}&\mathrm{Hier}(t, \mathcal{T}_c, \mathcal{C}) = \\ &\forall v, w \text{ nodes of } t, \quad (v, w) \in \mathcal{O}_c^t \Longleftrightarrow \big(\mathcal{C}(v), \mathcal{C}(w)\big) \in \mathcal{O}_c.\end{aligned}$$

Hence, the probability density for the observed tree t given the model $\mathcal{T}_c$ and the correspondences $\mathcal{C}$ is

$$P(t|\mathcal{T}_c, \mathcal{C}) = \begin{cases} \prod_{j \in \mathcal{N}^t} \theta_{c,\mathcal{C}(j)} \prod_{i \in \mathcal{N}_c \setminus \mathrm{Im}(\mathcal{C})} (1 - \theta_{c,i}) & \text{if } \mathrm{Hier}(t, \mathcal{T}_c, \mathcal{C}) \\ 0 & \text{otherwise.} \end{cases}$$

The image of the correspondence map $\mathcal{C}$, i.e., $\mathrm{Im}(\mathcal{C})$, is the set of model nodes that appear in the observed tree t with node-set $\mathcal{N}^t$.

2.2 Model Coding

Let $\mathcal{M}$ be a k-dimensional parametric model, the MDL code-length for a m-dimensional sample vector $\bar{x}$ is

$$\mathrm{LL}(\bar{x}|\mathcal{M}) = -\log(P(\bar{x}|\hat{\Theta}(\bar{x}))) + \frac{k}{2}\log(m),$$

where $\hat{\Theta}(\bar{x})$ is the maximum-likelihood estimate of the parameters of $\mathcal{M}$. The code-length can be divided into the cost of describing the data $\mathrm{LL}(\bar{x}|\hat{\Theta}(\bar{x})) = -\log(P(\bar{x}|\hat{\Theta}(\bar{x})))$, and the cost of describing the model parameters $\mathrm{LL}(\hat{\Theta}(\bar{x})) = \frac{k}{2}\log(m)$.

Given our tree-model, the cost of describing a tree t given the model $\mathcal{T}_c$ and the correspondences $\mathcal{C}$ is

$$-\log P(t|\mathcal{T}_c, \mathcal{C}) = \begin{cases} -\sum_{i \in \mathrm{Im}(\mathcal{C})} \log \theta_{c,i} - \sum_{j \in \mathcal{N} \setminus \mathrm{Im}(\mathcal{C})} \log(1 - \theta_{c,j}) & \text{if } \mathrm{Hier}(t, \mathcal{T}_c, \mathcal{C}) \\ \infty & \text{otherwise.} \end{cases}$$

Hence, the cost of describing the data set $\mathcal{D} = \{t_1, t_2, \ldots, t_N\}$ using the mixture model $\mathcal{H} = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_k\}$ is

$$\begin{aligned}\mathrm{LL}(\mathcal{D}|\mathcal{H}, \mathcal{C}) &= -\log P(\mathcal{D}|\mathcal{H}, \mathcal{C}) = -\sum_{t \in \mathcal{D}} \log P(t|\mathcal{H}, \mathcal{C}) \\ &= -\sum_{t \in \mathcal{D}} \log \left[\sum_{c=1}^{k} \alpha_c P(t|\mathcal{T}_c, \mathcal{C}) \right].\end{aligned}$$

Assume that the tree t can arise as an observation from only one tree-model $\mathcal{T}_c$. We can encode this association using an indicator variable $\vec{z}$ that links each observed data tree to one model. The indicator takes on the values

$$z_c^t = \begin{cases} 1 & \text{if tree } t \text{ is observed from model } \mathcal{T}_c \\ 0 & \text{otherwise.} \end{cases}$$

Including the indicator variable, the cost of describing the observed data given the model is

$$\begin{aligned}\mathrm{LL}(\mathcal{D}|\vec{z}, \mathcal{H}, \mathcal{C}) &= -\sum_{t \in \mathcal{D}} \sum_{c=1}^{k} z_c^t \log P(t|\mathcal{T}_c, \mathcal{C}) \\ &= -\sum_{c=1}^{k} \sum_{t \in \mathcal{D}_c} \log P(t|\mathcal{T}_c, \mathcal{C}),\end{aligned}$$

where $\mathcal{D}_c = \{t \in \mathcal{D} | z_c^t = 1\}$ is the set of samples generated by the model indexed c, i.e., $\mathcal{T}_c$. To this cost, we need to add that incurred in encoding the observation probabilities $\hat{\Theta}_c$, i.e.,

$$\mathrm{LL}(\hat{\Theta}_c|\vec{z}, \mathcal{H}, \mathcal{C}) = \sum_{c=1}^{k} \frac{n_c}{2} \log(m_c),$$

where n_c is the number of nodes in model $\mathcal{T}_c$ and $m_c = \sum_{t\in\mathcal{D}} z_c^t$ is the number of trees assigned to the model. Further, the cost of encoding the correspondence mapping between the observed data-trees and the model that generated them is:

$$\mathrm{LL}(\vec{z}|\mathcal{H}, \mathcal{C}) = -\sum_{c=1}^{k} \log \alpha_c \sum_{t\in\mathcal{D}} z_c^t.$$

As for the model encoding, we make the simplifying assumption that all models with n nodes and their feasible correspondences are equally likely. It is well-known [27] that the number of ordered trees with n nodes is C_{n-1} where $C_n = \frac{1}{n+1}\binom{2n}{n}$ is the nth Catalan number. Asymptotically, we have $C_n \approx 4^n$. We also assume an exponential prior over the model dimension, yielding a model encoding cost of $\mathrm{LL}(\mathcal{T}|\mathcal{C}) = n(2+\lambda) + const.$, where λ is the prior term. Here, we deal with unordered trees, so the previous equation overestimates the cost. The number of unordered trees is given by the Wedderburn-Etherington numbers [40]. An explicit formula for this sequence is elusive, but its asymptotic behavior is exponential. This leads to the model encoding cost $\mathrm{LL}(\mathcal{T}|\mathcal{C}) = n(\zeta+\lambda) + const.$, with $\zeta \approx 1.31$. By setting $g = \zeta + \lambda$, we obtain the cost of describing the full model $\mathcal{H}$

$$\mathrm{LL}(\mathcal{H}|\mathcal{C}) = g\sum_{c=1}^{k} n_c + \frac{k-1}{2}\log(m) + const.,$$

where n_c is the number of nodes in model $\mathcal{T}_c$ and $\frac{k-1}{2}\log(m)$, with $m = |\mathcal{D}|$, is the cost of describing the mixing parameters $\bar{\alpha}$. Since the constant part does not affect the solution, we will discard it from further consideration.

Hence, by adding together the different contributions, the description cost is

$$\mathrm{LL}(\mathcal{D}, \mathcal{H}|\mathcal{C}) = \sum_{c=1}^{k}\left[-\sum_{t\in\mathcal{D}_c}\log P(t|\mathcal{T}_c, \mathcal{C}) + n_c\left(\frac{\log(m_c)}{2} + g\right) - m_c \log\alpha_c\right] + \frac{k-1}{2}\log(m). \quad (2)$$

Let us return to the example presented in Fig. 1. Assume that we have a sample of n trees generated by the tree model, and that we wish to infer the structural model that generated them. Since we do not have knowledge about the correspondences, we will see only three possible tree structures in our data, and these occur with relative frequencies 0.25, 0.5, and 0.25, respectively (see Fig. 2). From this data, the model depicted in Fig. 2 can be inferred. There are multiple sources of bias in this estimation process that conspire to produce the observed result. First, the use of the proposed three-part code, which is not a universal code, does not take into account the stochastic complexity of the models, nor the number of distinct observations that can be made from the same model. Overcoming this would be problematic since it would require the computation of the dimension of the isomorphic group of the structural model, that is, the group of all possible isomorphisms, i.e., node-correspondences from the structural model to itself. However, this source of bias becomes increasingly weaker as the number of samples increases. Second, there is a bias induced by the ML estimation of the node-correspondences. By using a single ML estimate of the correspondences, we do not take into account that a model can produce the same observed tree in different ways. That is, there can be more than one set of correspondences that gives rise to the same observation. This source of bias is, in a sense, complementary to the previous one. The reason for this is that it wrongly penalizes models which have a large isomorphic group, whereas the first source of bias fails do to so when it should. Furthermore, this second source of bias is arguably more relevant since it does not vanish asymptotically. Overcoming this source of bias is problematic since it would require that the data code-length is averaged over all sets of correspondences that are consistent with the hierarchical constraint. Work is currently underway aimed at overcoming both sources of bias for the general graph estimation problem. We hope to do this by employing Bayesian approach to unify both model coding and correspondence estimation. However, the computational complexity of the underlying problem means that goal specific approximations must be used.

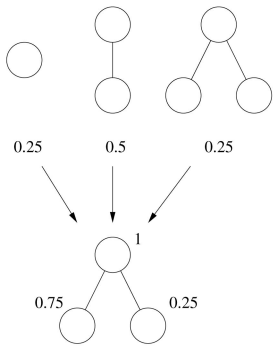

Fig. 2. Estimating a tree model from samples without correspondences.

2.3 Maximum-Likelihood Node Parameter Estimation

To estimate parameters of our model, i.e., the node sampling probabilities, we compute the log-likelihood of the sample data $\mathcal{D}_c$ given the tree-union model $\mathcal{T}_c$ and the correspondence mapping function $\mathcal{C}$. Under the assumption that the sampling process acts independently on the nodes of the structure, the log-likelihood is

$$\mathcal{L}(\mathcal{D}|\vec{z}, \mathcal{H}, \mathcal{C}) = \sum_{c=1}^{k} \mathcal{L}(\mathcal{D}_c|\mathcal{T}_c, \mathcal{C}),$$

where $\mathcal{L}(\mathcal{D}_c|\mathcal{T}_c, \mathcal{C}) = \sum_{t\in\mathcal{D}_c}\log P(t|\mathcal{T}_c, \mathcal{C})$ is the per-component log-likelihood.

Let $K_{c,i}^t = \{j \in \mathcal{N}^t | C(j) = i\}$ be the set of nodes in the different trees in $\mathcal{D}_c$ for which $\mathcal{C}$ maps a node to $i \in \mathcal{N}_c$. Since the node mapping is one-to-one, $K_{c,i}^t$ is either the empty set or the singleton $\{j\}$ with $C(j) = i$. Further, let

$l_{c,i} = \sum_{t \in \mathcal{D}_c} |K^t_{c,i}|$ be the number of trees in $\mathcal{D}_c$ that map a node to i, and $m_c = |\mathcal{D}_c|$ be the number of trees assigned to model $\mathcal{T}_c$. Since we are using a simple Bernoulli model, the ML estimate of the sampling probability is $\theta_{c,i} = \frac{l_{c,i}}{m_c}$. When the estimates of the sampling probabilities are substituted into the log-likelihood function, we have that

$$\begin{aligned}\hat{\mathcal{L}}(\mathcal{D}_c|\mathcal{T}_c,\mathcal{C}) &= \sum_{i\in\mathcal{N}_c} m_c \left[\frac{l_i}{m_c}\log\left(\frac{l_i}{m_c}\right) + \left(1-\frac{l_i}{m_c}\right)\log\left(1-\frac{l_i}{m_c}\right)\right] \\ &= -\sum_{i\in\mathcal{N}_c} m_c I(\theta_{c,i}),\end{aligned} \tag{3}$$

where $I(\theta_{c,i}) = -\big[\theta_{c,i}\log(\theta_{c,i}) + (1-\theta_{c,i})\log(1-\theta_{c,i})\big]$ is the entropy of the probability distribution for sampling node i from class c. This equation holds provided that there exists an order relation that is respected by every hierarchical tree in the sample set $\mathcal{D}_c$. If this is not the case, then the log-likelihood function takes on the value $\hat{\mathcal{L}}(\mathcal{D}_c|\mathcal{T}_c,\mathcal{C}) = -\infty$.

Similarly, the mixing proportions $\bar{\alpha} = \{\alpha_c; c = 1,\ldots,k\}$, are estimated using the observed frequency ratio $\alpha_c = \frac{m_c}{m}$. With these estimates, the code-length becomes:

$$\mathrm{LL}(\mathcal{D},\mathcal{H}|\mathcal{C}) = \sum_{c=1}^{k}\sum_{i\in\mathcal{N}_c} \mathrm{LL}^i(\mathcal{D}_c,\mathcal{T}_c|\mathcal{C}) + mI(\bar{\alpha}) + \frac{k-1}{2}\log(m), \tag{4}$$

where

$$\mathrm{LL}^i(\mathcal{D}_c,\mathcal{T}_c|\mathcal{C}) = m_c I(\theta_{c,i}) + \frac{\log(m_c)}{2} + g \tag{5}$$

is the per-node description cost, and $I(\bar{\alpha}) = -\sum_{c=1}^{k}\frac{m_c}{m}\log\left(\frac{m_c}{m}\right)$.

3 Weighted Model

We can extend the method outlined in the previous section to the case where the nodes have weights. When this is the case, then we would expect the sampling likelihood to reflect the distribution of node weights. Hence, the simple probability distribution described above and which is based on uniform sample node probability is not sufficient because it does not take into account the weight distribution. To overcome this shortcoming, in addition to the set of sampling probabilities Θ_c, we associate with the union model a weight distribution function $\mathcal{W}_c$. In general, Θ_c and $\mathcal{W}_c$ are not independent. In fact, we would expect the most relevant nodes to be associated with larger weights and to be sampled with higher probability. In particular, we model the mutual dependency by deriving $\theta_{c,i}$ and $W_{c,i}$ from a single stochastic node-observation model $X_{c,i}$. The idea behind the node-observation model is that the each node is observed only if it provides a sufficiently strong signal. We assume that $X_{c,i}$ is normally distributed with mean $\mu_{c,i}$ and standard deviation $\sigma_{c,i}$. As a result, the probability distribution of a sample x from $X_{c,i}$ is

$$\frac{1}{\sigma_{c,i}\sqrt{2\pi}}\exp\left(-\frac{1}{2}\frac{(x-\mu_{c,i})^2}{\sigma^2_{c,i}}\right).$$

When sampling a tree from the tree model $\mathcal{T}_c$, for each node i, we select a sample x_i from $X_{c,i}$. If $x_i \geq 0$, node i will be observed in the tree and with a weight $w_i = x_i$. Conversely, if $x_i < 0$, node i will not be present in the tree sample. Hence, the weight distribution $W_{c,i}$ will follow the probability distribution

$$P_w(w|\mu_{c,i},\sigma_{c,i}) = \begin{cases} \frac{1}{\theta_{c,i}\sigma_{c,i}\sqrt{2\pi}}\exp\left(-\frac{1}{2}\frac{(w-\mu_{c,i})^2}{\sigma^2_{c,i}}\right) & \text{if } w \geq 0 \\ 0 & \text{otherwise.}\end{cases}$$

The sampling probability is the integral of the distribution over positive weights, i.e.,

$$\theta_{c,i} = \int_0^\infty \frac{\exp\left(-\frac{1}{2}\frac{(w-\mu_{c,i})^2}{\sigma^2_{c,i}}\right)}{\sigma_{c,i}\sqrt{2\pi}}dw = 1 - \mathrm{erfc}(\tau_{c,i}), \tag{6}$$

where $\tau_{c,i} = \mu_{c,i}/\sigma_{c,i}$ and erfc is the complementary error function

$$\mathrm{erfc}(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}s^2\right)ds.$$

An advantage of using this observation model for learning tree structure is that it provides a direct indication of whether a node is a genuine feature or simply noise. In fact, the larger (positive) $\tau_{c,i}$, the higher the confidence that the node is a feature. Furthermore, the distribution of weight of the samples extracted from the node has a maximum value at the mean signal-strength $\mu_{c,i}$. Conversely, if $\tau_{c,i}$ is negative, the node is most likely to be due to noise and the weight distribution will decrease monotonically with w.

In the weighted case, the complete structural model is represented by the tuple $\mathcal{T}_c = (\mathcal{N}_c, \mathcal{O}_c, \bar{\tau}_c, \bar{\sigma}_c)$, where $\bar{\tau}_c = \{\tau_{c,i}; i \in \mathcal{N}_c\}$ and $\bar{\sigma}_c = \{\sigma_{c,i}; i \in \mathcal{N}_c\}$ are sets of node parameters.

According to this weighted model, model $\mathcal{T}_c$ generates tree t with probability

$$P_w(t|\mathcal{T}_c,\mathcal{C}) = \begin{cases} \prod_{j\in\mathcal{N}^t}\theta_{c,\mathcal{C}(j)}P(w_j|\mu_{c,\mathcal{C}(j)},\sigma_{c,\mathcal{C}(j)}) \prod_{i\in\mathcal{N}_c\setminus\mathrm{Im}(\mathcal{C})}(1-\theta_{c,i}) & \text{if } \mathrm{Hier}(t,\mathcal{T}_c,\mathcal{C}) \\ 0 & \text{otherwise.}\end{cases}$$

Using the weighted model there are two differences in the description cost. First, there is the change in the observation probability and, hence, on the cost of describing the data given the model parameters. The expression for the cost is

$$\mathrm{LL}_w(\mathcal{D}|\vec{z},\mathcal{H},\mathcal{C}) = -\sum_{c=1}^{k}\sum_{t\in\mathcal{D}_c}\log P_w(t|\mathcal{T}_c,\mathcal{C}),$$

where $P_w(t|\mathcal{T}_c,\mathcal{C})$ is the probability of the weighted observation model. The second difference derives from the added complexity of the model. Specifically, the model now has two parameters per node, instead of just one parameter in the unweighted case. Hence, the cost of describing the parameters is now

$$\mathrm{LL}_w(\hat{\tau},\hat{\sigma}|\vec{z},\mathcal{H},\mathcal{C}) = \sum_{c=1}^{k} n_c\log(m_c).$$

All the remaining contributions remain unchanged, yielding the description cost

$$\begin{aligned}&\mathrm{LL}_w(\mathcal{D},\mathcal{H}|\mathcal{C}) = \\ &\sum_{c=1}^{k}\Bigg[-\sum_{t\in\mathcal{D}_c}\log P_w(t|\mathcal{T}_c,\mathcal{C}) \\ &+ n_c(\log(m_c)+g) - m_c\log\alpha_c\Bigg] + \frac{k-1}{2}\log(m).\end{aligned} \tag{7}$$

3.1 Maximum-Likelihood Parameters

Using the *weighted* node model, the log-likelihood function cannot be expressed as the sum of the per-node entropies. However, it can be expressed as the sum of per-node log-likelihood functions

$$\begin{aligned}\mathcal{L}_w(\mathcal{D}_c|\mathcal{T}_c,\mathcal{C}) = \sum_{i\in\mathcal{N}_c}\Bigg[&(m_c - l_{c,i})\log(\mathrm{erfc}(\tau_{c,i})) \\ &-\frac{l_{c,i}}{2}\log(2\pi\sigma_{c,i}) - \frac{1}{2}\sum_{t\in\mathcal{D}_c}\sum_{j\in K^t_{c,i}}\left(\frac{w^t_j}{\sigma_{c,i}} - \tau_{c,i}\right)^2\Bigg],\end{aligned}$$

where w^t_j is the weight of node j of tree t, $K^t_{c,i} = \{j \in \mathcal{N}^t | C(j) = i\}$ is the set of nodes in the different trees in $\mathcal{D}_c$ for which $\mathcal{C}$ maps a node to $i \in \mathcal{N}_c$, $m_c = |\mathcal{D}_c|$ is the number of trees in $\mathcal{D}_c$, and $l_{c,i} = \sum_{t\in\mathcal{D}_c}|K^t_{c,i}|$ is the number of times node i is observed in $\mathcal{D}_c$.

To estimate the parameters of the weight distribution, we take the derivatives of the log-likelihood function with respect to σ_i and τ_i and set them to zero. As a result,

$$\sigma_{c,i} = -\frac{\tau_{c,i}}{2}\overline{W_{c,i}} + \sqrt{\left(\frac{\tau_{c,i}}{2}\overline{W_{c,i}}\right)^2 + \overline{W^2_{c,i}}} \tag{8}$$

$$(m_c - l_{c,i})\mathrm{erfc}'(\tau_{c,i}) + l_{c,i}\mathrm{erfc}(\tau_{c,i})\left(\frac{\overline{W_{c,i}}}{\sigma_{c,i}} - \tau_{c,i}\right) = 0, \tag{9}$$

with

$$\overline{W_i} = \frac{1}{l_{c,i}}\sum_{t\in\mathcal{D}_c}\sum_{j\in K^t_{c,i}} w^t_j$$

and

$$\overline{W^2_i} = \frac{1}{l_{c,i}}\sum_{t\in\mathcal{D}_c}\sum_{j\in K^t_{c,i}}\left(w^t_j\right)^2.$$

It is clear that when $l_{c,i} = m_c$ then the likelihood function is maximized by

$$\sigma_{c,i} = \sqrt{\overline{W^2_{c,i}} - \overline{W_{c,i}}^2}$$

and $\tau_{c,i} = \frac{\overline{W_{c,i}}}{\sigma_{c,i}}$. When $l_{c,i} < m_c$, we maximize the log likelihood by setting ${\tau_{c,i}}^{(0)} = \mathrm{erfc}^{-1}(\frac{m_c - l_{c,i}}{m_c})$ and iterating the recurrence:

$${\sigma_{c,i}}^{(k)} = -\frac{{\tau_{c,i}}^{(k)}}{2}\overline{W_{c,i}} + \sqrt{\left(\frac{{\tau_{c,i}}^{(k)}}{2}\overline{W_{c,i}}\right)^2 + \overline{W^2_{c,i}}} \tag{10}$$

$${\tau_{c,i}}^{(k+1)} = {\tau_{c,i}}^{(k)} - \frac{f({\tau_{c,i}}^{(k)}, {\sigma_{c,i}}^{(k)})}{\frac{d}{d{\tau_{c,i}}^{(k)}} f({\tau_{c,i}}^{(k)}, {\sigma_{c,i}}^{(k)})}, \tag{11}$$

where

$$\begin{aligned}&f({\tau_{c,i}}^{(k)}, {\sigma_{c,i}}^{(k)}) = \\ &(m_c - l_{c,i})\mathrm{erfc}'({\tau_{c,i}}^{(k)}) + l_{c,i}\mathrm{erfc}({\tau_{c,i}}^{(k)})\left(\frac{\overline{W_{c,i}}}{{\sigma_{c,i}}^{(k)}} - {\tau_{c,i}}^{(k)}\right).\end{aligned}$$

Substituting the estimates and dropping the iteration index, we obtain the cost

$$\begin{aligned}\mathrm{LL}_w(\mathcal{D},\mathcal{H}|\mathcal{C}) = &\sum_{c=1}^{k}\sum_{i\in\mathcal{N}_c}\mathrm{LL}^i_w(\mathcal{D}_c,\mathcal{T}_c|\mathcal{C}) \\ &+ mI(\bar{\alpha}) + \frac{k-1}{2}\log(m),\end{aligned} \tag{12}$$

where the per-node description cost is

$$\begin{aligned}\mathrm{LL}^i_w(\mathcal{D}_c,\mathcal{T}_c|\mathcal{C}) = &-(m_c - l_{c,i})\log(\mathrm{erfc}(\tau_{c,i})) + \frac{l_{c,i}}{2}\log(2\pi\sigma_{c,i}) \\ &+\frac{1}{2}\sum_{t\in\mathcal{D}_c}\sum_{j\in K^t_{c,i}}\left(\frac{w^t_j}{\sigma_{c,i}} - \tau_{c,i}\right)^2 + \log(m_c) + g.\end{aligned} \tag{13}$$

4 Learning the Mixture

Our aim is to learn the set of tree-union models that partition the training data $\mathcal{D}$ into nonoverlapping classes $\mathcal{D}_1, \cdots, \mathcal{D}_k$ so as to minimize the description cost. Unfortunately, locating the global minimum of the description length in this way is an intractable combinatorial problem. There are two interrelated optimization problems that must be solved. The first is that of estimating the tree-unions given the class assignment indicators $\bar{z}$. The second problem is that of estimating the class assignment indicators $\bar{z}$. One popular way of solving problems of this sort is to use the Expectation-Maximization algorithm [5], and to treat the class indicators as missing data. However, it is not easily used here due to the structural nature of the class tree union models. Since the tree unions are found by recursively merging sample trees, they are data-dependent pattern space models. The domain of the optimization process depends on the tree samples assigned to the clusters. Hence, we resort to a local search technique, whose main requirement is that we can optimally merge two tree models. The approach is as follows:

- Commence with an overly specific model. We use a structural model per sample-tree, where each model is equiprobable and structurally identical to the respective sample-tree, and each node has unit sample probability.
- Iteratively generalize the model by merging pairs of tree-unions. The candidates for merging are chosen so that they maximally decrease the description length.
- The algorithm stops when there are no merges remaining that can decrease the description length.

This is clearly a variant of the hierarchical clustering method outlined in [18] in which the topmost level and, hence, the composition of the extracted clusters is controlled by the description length criterion.

We merge two tree models by calculating their tree-union that is a superstructure that has the property that the original trees can be obtained from it using a sequence of

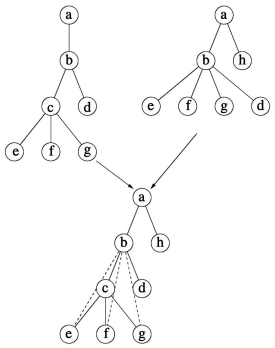

Fig. 3. The new model is obtained by merging matching nodes.

node removal operations (see Fig. 3). This means that the hierarchical constraints of the original models and, hence, of all the trees assigned to them is preserved. This, in turn, means that all the trees assigned to the merged model are valid observations of the model.

Given two tree models $\mathcal{T}_1$ and $\mathcal{T}_2$, we wish to construct a union $\hat{\mathcal{T}}$ whose structure respects the hierarchical constraints present in both $\mathcal{T}_1$ and $\mathcal{T}_2$, and that also minimizes the description cost for the merged model $\hat{\mathcal{T}}$ and data $\hat{\mathcal{D}} = \mathcal{D}_1 \cup \mathcal{D}_2$, where $\mathcal{D}_1$ and $\mathcal{D}_2$ are the sample sets used to learn $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively. Since the trees $\mathcal{T}_1$ and $\mathcal{T}_2$ already assign node correspondences $\mathcal{C}_1$ and $\mathcal{C}_2$ from the data samples to the model, we can simply find a map $\mathcal{M}$ from the nodes in $\mathcal{T}_1$ and $\mathcal{T}_2$ to $\hat{\mathcal{T}}$ and transitively extend the correspondences from the samples in $\hat{\mathcal{D}}$ to the final model $\hat{\mathcal{T}}$ in such a way that, given two nodes $v \in \mathcal{N}_1$ and $v' \in \mathcal{N}_2$, then $\hat{\mathcal{C}}(v) = \hat{\mathcal{C}}(v') \Leftrightarrow v' = \mathcal{M}(v)$.

Posed as the merge of two structures, the correspondence problem is reduced to that of finding the set of nodes in $\mathcal{T}_1$ and $\mathcal{T}_2$ that are common to both trees. For each pair of nodes, we consider the two alternative hypotheses. The first is that the nodes match, and this strengthens the confidence that the nodes correspond to a feature of the model. The second hypothesis is that the two nodes do not match, and this reinforces the confidence that they represent structural noise. We compare the description cost incurred in by the alternative hypotheses. Starting with the two structures, we merge the sets of nodes that reduces the description length by the largest amount, while still satisfying the hierarchical constraint. That is we merge nodes u and v of $\mathcal{T}_1$ with node u' and v' of $\mathcal{T}_2$, respectively, if and only if $(u, v) \in \mathcal{O}_1 \Leftrightarrow (u', v') \in \mathcal{O}_2$, where $\mathcal{O}_1$ and $\mathcal{O}_2$ are the tree-order relations of models $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively. This generalization process allows us to circumvent the problem of directly modeling the detailed probability distribution for structural noise. Instead, we learn the distribution from the data samples.

From (4) and (12), we see that the description length is linear with respect to the contribution from each node of each component of the mixture. This allows us to pose the minimization of the description length as a linear optimization problem with a combinatorial constraint. In particular, as we will show in the next section, we can pose the model-merging problem as an instance of a particular minimum tree edit-distance problem.

Let $m_1 = |\mathcal{D}_1|$ and $m_2 = |\mathcal{D}_2|$ be the number of tree samples from $\hat{\mathcal{D}}$ that are respectively assigned to $\mathcal{T}_1$ and $\mathcal{T}_2$. Further, let l_v and $l_{v'}$ be the number of times the nodes v and v' in $\mathcal{T}_1$ and $\mathcal{T}_2$ are observed in the trees in $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. With the *unweighted* model, if the two nodes are not merged, then the sampling probabilities are $\theta_v = \frac{l_v}{m_1+m_2}$ and $\theta_{v'} = \frac{l'_v}{m_1+m_2}$, respectively, while the sampling probability of the merged node is $\theta_{(vv')} = \frac{l_v+l_{v'}}{m_1+m_2}$. Hence, the description length advantage obtained by merging the nodes v and v' is

$$\begin{aligned}\mathcal{A}(v, v') &= \mathrm{LL}^{v}(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C}) + \mathrm{LL}^{v'}(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C}) - \mathrm{LL}^{vv'}(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C}) \\ &= (m_1 + m_2)\big[I(\theta_v) + I(\theta_{v'}) - I(\theta_{(vv')})\big] \\ &\quad + \frac{1}{2}\log(m_1 + m_2) + g.\end{aligned} \tag{14}$$

When using the *weighted* model the advantage is:

$$\begin{aligned}\mathcal{A}_w(v, v') &= \mathrm{LL}_w^{v}(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C}) \\ &\quad + \mathrm{LL}_w^{v'}(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C}) - \mathrm{LL}_w^{vv'}(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C}).\end{aligned} \tag{15}$$

In both the unweighted or the weighted cases, the set of merges $\mathcal{M}$ that minimizes the description length of the combined tree-union also maximizes the advantage function

$$\mathcal{A}(\mathcal{M}) = \sum_{(v,v') \in \mathcal{M}} \mathcal{A}(v, v'). \tag{16}$$

At the end of the node merging operation, we have a set of nodes that respects the partial order relations present in both the original models. The order relation is hence respected by all the sample-trees in the merged sample set $\hat{\mathcal{D}}$. The new merged set of nodes, order relation and estimated node parameters define a new tree model $\hat{\mathcal{T}}$. Among all possible pairs of model merges, T_1 and T_2 is the one that minimizes the description length. Finally, the learning algorithm is as follows:

1. Initialize the algorithm by creating one mixture component per tree-sample in $\mathcal{D}$ and calculate the description length of the resulting model.
2. For each pair of initial mixture components, calculate their union and the description length of the merged structure. The mixing proportion for this optimal merge is equal to the sum of the proportions of the individual unions.
3. From the set of all potential merges, identify the one which reduces the description cost by the greatest amount. If the description cost increases for all merges, stop the algorithm.
4. Calculate the union and description cost that results from merging the newly obtained model with each of the remaining components.
5. Goto 3 and repeat until the description length cannot be reduced any further.

To conclude this section, we refer to Fig. 4 which illustrates an example of the merge of six sample trees. The figure shows the structural archetype of the merged models after each stage. The shading of the nodes represents their sampling probabilities, and the higher the probability, the darker the node.

5 Tree Edit-Distance

As noted in earlier, the description length advantage is related to the edit distance between tree structures. This is an important observation. One of the difficulties with graph

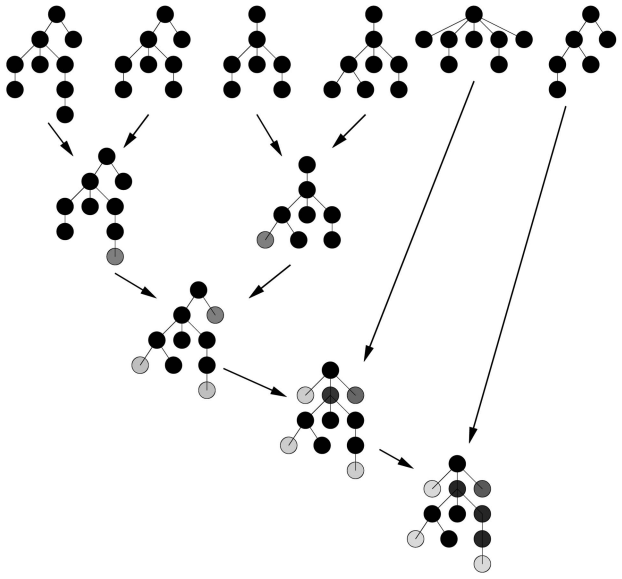

Fig. 4. Merging sample trees into a single tree-model.

edit distance [31], [6] is that there is no clear methodology for assigning costs to edit operations. By contrast, in the work reported in this paper the description length changes associated with tree merge operations are determined by the node probabilities, and these, in turn, may be estimated from the available tree-samples. By establishing a link between edit-distance and description length, we provide a means by which edit costs may be estimated.

Hence, in this section, we review the computation of tree edit-distance developed in our previous work [37]. In particular, we describe how tree edit distance may be used to estimate node-correspondences and give an overview of the algorithm we use to approximate the computation of tree edit distance.

The idea behind edit distance is that it is possible to identify a set of basic edit operations on nodes and edges of a structure, and to associate with these operations a cost. The edit-distance is found by searching for the sequence of edit operations that will make the two graphs isomorphic with one-another and which have minimum cost. The optimal sequence can be found using only structure reducing operations. This can be explained by the fact that we can transform node insertions in one tree into node removals in the other. This means that the edit distance between two trees is completely determined by the subset of residual nodes remaining after the optimal removal sequence or, equivalently, by the nodes that are in correspondence. This means that the constraints imposed by the edit-distance framework on the set of matching nodes are equivalent to those required to merge nodes on the model archetypes. Namely, that they preserve the hierarchy present in the two original structures.

The edit-distance between trees t and t' can be defined in terms of the nodes matched by the optimal correspondence $\mathcal{M}$:

$$D^{edit}(t,t') = \sum_{u \notin \mathrm{Dom}(\mathcal{M})} r_u + \sum_{v \notin \mathrm{Im}(\mathcal{M})} r_v + \sum_{(u,v)\in\mathcal{M}} m_{u,v}. \quad (17)$$

Here, r_u and r_v are the costs of removing nodes u and v, respectively, $m_{u,v}$ is the cost of matching u to v, and $\mathrm{Dom}(\mathcal{M})$ and $\mathrm{Im}(\mathcal{M})$ are the domain and image of the relation $\mathcal{M}$. Letting $\mathcal{N}^t$ be the set of nodes of tree t, the distance can be rewritten as:

$$D^{edit}(t,t') = \sum_{u\in\mathcal{N}^t} r_u + \sum_{v\in\mathcal{N}^{t'}} r_v + \sum_{(u,v)\in\mathcal{M}} (m_{uv} - r_u - r_v).$$

Hence, the distance is minimized by the set of correspondences that maximize the utility

$$\mathcal{U}(\mathcal{M}) = \sum_{(u,v)\in\mathcal{M}} (r_u + r_v - m_{uv}). \quad (18)$$

We can identify information theoretic node removal and merge costs by equating $\mathcal{U}$ above with the description length advantage $\mathcal{A}$. The costs that allow the *unweighted* problem to be posed as an edit distance problem are $r_v = (m_1 + m_2)I(\theta_v) + \frac{1}{2}\log(m_1+m_2) + g$ for the removal of node v, and $m_{vv'} = (m_1+m_2)I(\theta_{vv'}) + \frac{1}{2}\log(m_1+m_2) + g$ for matching node v with node v'. In the *weighted* case, the corresponding edit costs are $r_v = \mathrm{LL}_w^v(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C})$ for the removal of node v and $m_{vv'} = \mathrm{LL}_w^{vv'}(\hat{\mathcal{D}}, \hat{\mathcal{T}}|\mathcal{C})$ for matching node v with node v'.

Hence, our union-tree approach can be viewed as a means of learning tree-edit costs. This has been a long-standing problem since Fu et al. introduced the idea of graph edit distance in the early 1980s [31], [6].

In our experimental analysis, we used the method described in [37] to find the set of correspondences $\mathcal{M}^* = \arg\min_{\mathcal{M}} \mathcal{U}(\mathcal{M})$ that approximately minimize the edit distance.

6 Pattern Spaces from Union Trees

It is important to note that the mixture of weighted tree-unions also provides a route to embedding shapes of the same class in a pattern space. To do this, we use the correspondences with the union tree to map each tree onto a pattern vector. The components of the vector are unity if the corresponding sample has a corresponding node, and zero otherwise. We perform principal components analysis for the sample trees assigned to each class. To do this, we compute the covariance matrix for the pattern vectors and project the pattern vectors onto the space spanned by the leading eigenvectors of the covariance matrix.

We place the nodes of the union tree $\mathcal{T}_c$ in any arbitrary order. To each sample tree t we associate a pattern-vector $\vec{x}_t = (x_1, , x_n)^T \in \mathbb{R}^n$, where $n = |\mathcal{N}_c|$ is the number of nodes in the tree model $\mathcal{T}_c$. The component $x_t(i)$ of vector $\vec{x}$ is w_i^t if the tree t maps a node to node i of the model, 0 otherwise. In other words, we associate a pattern-vector $\vec{x}_t$ with the sample tree whose components are equal to the weight of the corresponding node in the union tree, if the node is present, and are zero otherwise. For each union-tree $\mathcal{T}_c$, we compute the mean pattern-vector $\bar{\vec{x}}_c = E[\vec{x}_t]$ (i.e., the expectation of $\vec{x}_t$), and its covariance matrix $\Sigma_c = E[(\vec{x}_t - \bar{\vec{x}}_c)(\vec{x}_t - \bar{\vec{x}}_c)^T]$. Suppose that the eigenvectors (ordered to decreasing eigenvalue) are $\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_{\mathcal{N}_c}$. The leading l_{sig} eigenvectors are used to form the columns of the matrix $E = (\vec{e}_1|\vec{e}_2|....|\vec{e}_{l_{sig}})$. We perform PCA on the sample-trees by projecting the pattern-vectors onto the leading eigenvectors of the covariance matrix. The projection of the pattern-vector for the sample tree indexed t is $\vec{y}_t = E^T\vec{x}_t$. The distance between the vectors in this space is $D^{PCA}(t,t') = (\vec{y}_t - \vec{y}_{t'})^T(\vec{y}_t - \vec{y}_{t'})$.

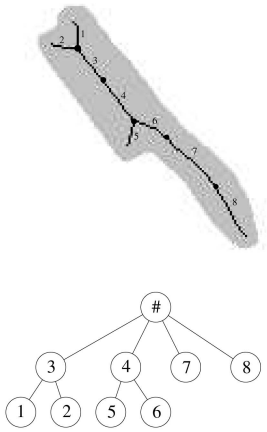

Fig. 5. A silhouette, its skeleton, and the corresponding shock-tree.

7 Experimental Results

We illustrate the utility of the tree-clustering algorithm on sets of shock trees. We commence from 2D shape silhouettes. For each shape, we extract the skeleton using the method outlined in [36]. The skeleton is the medial axis of the silhouette (i.e., the set of points that are equidistant from opposite object boundaries) and it is hence the locus of the center of a circle that is bitangent to the object boundary. The skeleton is segmented into branches along which the radius of the bitangent circle is either monotonically increasing or decreasing. Each branch segment is the vertex of a graph. Two vertices are connected by an edge if the associated branches are adjacent. The exception is when the point of branch contact is a minima of the radius of the bitangent circle. It is straightforward to show that this graph is a forest [33]. In order to transform the graph into a rooted tree, we add a dummy root-vertex that represents the barycenter of the silhouette. This dummy vertex is connected to the vertices that correspond to branches that are adjacent to points associated with a local maxima of the radius of the bitangent circle. Fig. 5 shows an example silhouette, with its skeleton superimposed, and the resulting shock-tree. Numeric labels correspond to the labels on the skeletal branches, while the dummy root is indicated by the symbol #. We consider two variants of the shock tree matching problem. In the first variant, we use a purely structural approach. The nodes in the shock trees are given uniform weight and we match only their structure. The second variant is weighted. Here, as outlined in [35], we assign to each shock group, that is, to each node i of shock tree t, a weight $w_i^t = ||l_1|| + ||l_2||$, where $||l_1||$ and $||l_2||$ are the lengths of the "left" and "right" boundary segments associated with the shock group. Our experiments are divided into three parts. We commence by illustrating qualitative examples of the clusters obtained with the two variants of our algorithm. This suggests that the weighted version is the most effective. We then focus in more detail on some of the quantitative properties of the weighted version. Finally, we provide a sensitivity analysis on synthetic data.

7.1 Clustering Examples

To illustrate the clustering process, we commence with a study on a small database of 25 shapes. In order to asses the quality of the method, we compare the clusters defined by the components of the mixture with those obtained by applying a graph spectral pairwise clustering method to the distances between graphs. The graph spectral clustering method is the maximum-likelihood technique developed by Robles-Kelly and Hancock [30]. This probabilistic method locates the clusters by iteratively extracting the eigenvectors from the matrix of edit-distances between the graphs so as to maximize a log-likelihood function. The edit distances are computed in two alternative ways. First, we compute weighted edit distance using the method outlined in Section 5. The second method involves computing the distance matrix using the projected tree-vectors obtained from PCA as described in Section 6.

Fig. 6 shows the clusters extracted from the database of 25 shapes. The first column shows the clusters extracted using the mixture of tree unions approach, and relies on a purely structural representation of shape. The second column displays the clusters extracted from the weighted edit-distances between shock-trees; here, the structural information is enhanced with geometrical information. The third column shows the clusters extracted from the distances obtained by embedding the geometrically enhanced shock-trees in a single tree-union. While there is some merge and leakage, the clusters extracted with the mixture of tree unions compare

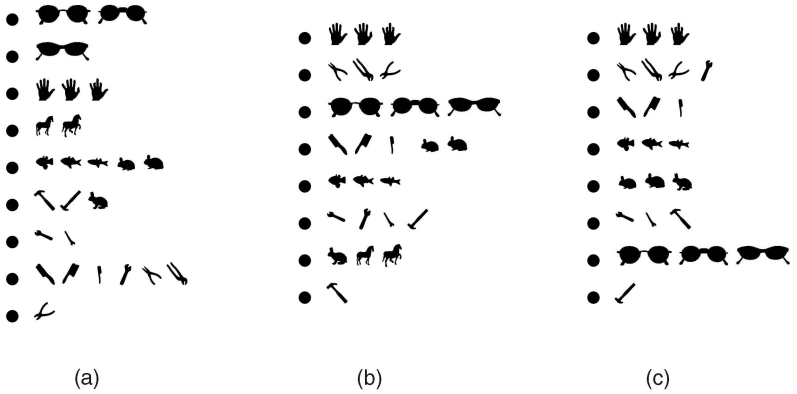

Fig. 6. Clusters extracted with a purely structural mixture of trees approach versus pairwise clustering of attributed distances obtained with edit distance and tree union. (a) Mixture of unattributed tree models. (b) Weighted edit-distance. (c) Union of attributed trees.

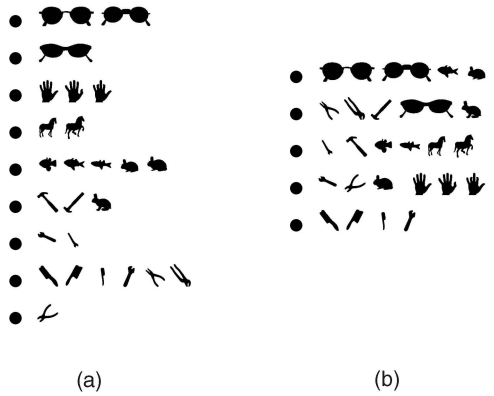

Fig. 7. Comparison of clusters obtained from nonattributed edit-distance and mixture of trees. (a) Mixture of tree models. (b) Pairwise clustering from edit-distance.

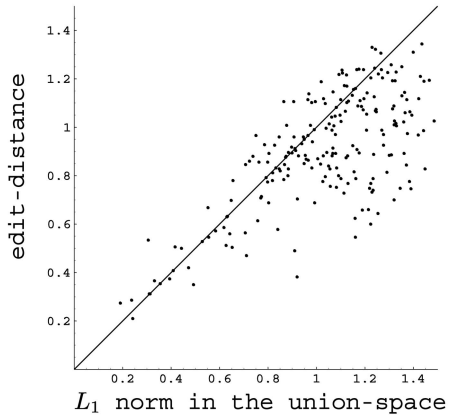

Fig. 8. Scatter plot of the distances computed using the edit-union versus the corresponding pairwise edit distance.

favorably with those obtained using the alternative clustering algorithms, even though these are based on data enhanced with geometrical information. The second to last cluster extracted using the mixture of tree unions deserves some further explanation. The structure of the shock-trees of the distinct tools in the cluster are identical. Hence, by using only structural information, the method clusters the shock-trees together. To distinguish between the objects, geometrical information must be provided too. Hence, the two alternative clustering methods are able to distinguish between the wrenches, brushes, and pliers.

Fig. 7 compares the mixture of tree unions, and the pairwise clustering method when applied to the unweighted edit distance between graphs. The clusters obtained using the mixture of tree-unions are shown on the left, while those obtained through the pairwise clustering of unweighted edit-distance are shown on the right. These results suggest that the mixture of tree-unions method outperforms pairwise clustering of edit-distance on purely structural data.

It is also interesting to consider the relationship between the edit distances defined on the union structure and the conventional edit distance. Fig. 8 plots the distances obtained using the union of weighted shock trees (x axis) versus the corresponding pairwise edit distances (y axis). The main feature to note from the plot is that the pairwise distance method tends to underestimate the distances between shapes.

Fig. 9. Two-dimensional multidimensional scaling of the pairwise distances of the shock graphs. The numbers correspond to the shape classes.

7.2 Quantitative Analysis

We now turn our attention to the properties of the weighted variant of our mixture of tree unions clustering method, when applied to a larger database of 120 trees. The database consists of 120 shapes divided into eight shape classes containing 15 shapes each.

To perform an initial evaluation of this database, we have applied multidimensional scaling to the weighted edit distances between the shock graphs for the different shapes. By doing this, we embed points representing the graphs in a low dimensional space spanned by the eigenvectors of a similarity matrix computed from the pairwise distances. In Fig. 9, we show the projection of the graphs onto the 2D space spanned by the leading two eigenvectors of the similarity matrix. Each label in the plot corresponds to a particular shape class. Label 1 identifies hands, label 2 horses, label 3 ducks, label 4 men, label 5 pliers, label 6 screwdrivers, label 7 dogs, and, finally, label 8 is associated with leaves. The plot clearly shows the difficulty of this clustering problem. The shape groups are not well separated. Rather, there is a good deal of overlap between them. Furthermore, there are a considerable number of outliers.

To assess the ability of the clustering algorithm to separate the shape classes, we performed experiments on an increasing number of shapes. We commenced with the 30 shapes from the first two shape classes and then increased the number of shape classes under consideration until the full set of 120 shapes was included. We compare the clusters obtained with the mixture of tree-unions approach with the result obtained by applying the pairwise clustering algorithm to two different distance measure. The first of these is approximate edit-distance described in [37], while the second is a tree distance metric that can be computed in polynomial time [38]. The distance metric between trees t_1 and t_2 is computed using the function $\sigma(u, v)$ that gauges the similarity between node u in t_1 and node v in t_2. The similarity measure for the two trees is $S(t_1, t_2) = \max_\phi \sum_{(u,v)\in\phi} \sigma(u, v)$, where ϕ is

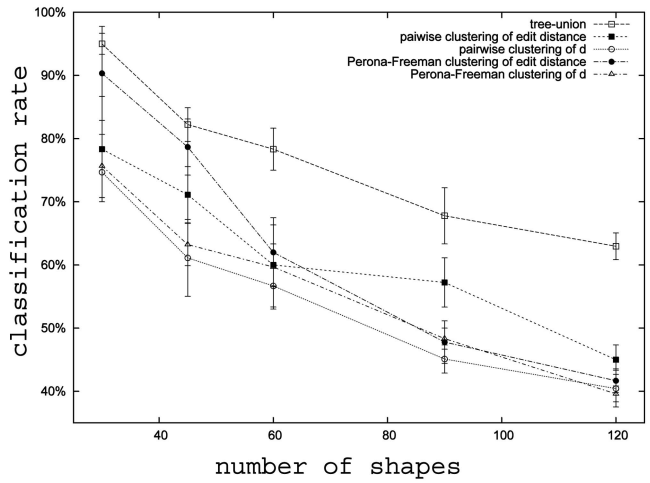

Fig. 10. Proportion of correct classifications obtained with the mixture of trees versus those obtained with pairwise clustering.

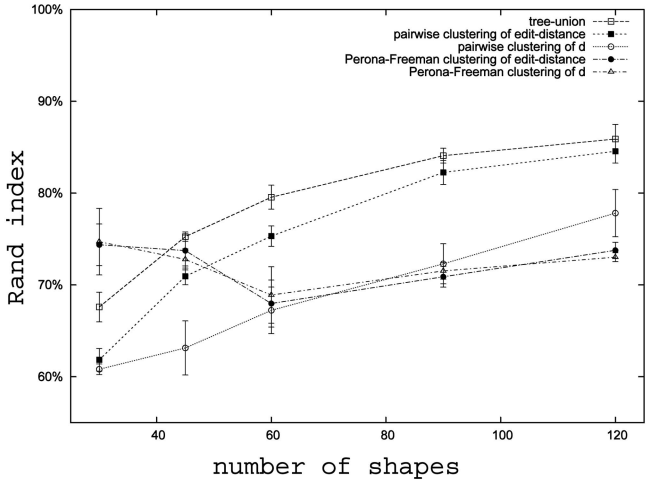

Fig. 11. Rand index obtained with the mixture of trees versus those obtained with pairwise clustering.

a subtree isomorphism between t_1 and t_2. Hence, the distance metric is defined as:

$$D^{metric}(t_1, t_2) = 1 - \frac{S(t_1, t_2)}{\max(|t_1|, |t_2|)}.$$

Additionally, we explore the use of two graph-spectral algorithms for clustering the different distance measures. The first is the maximum-likelihood method described earlier [30]. The second is the matrix factorization algorithm developed by Perona and Freeman in [28].

In order to assess the quality of the groupings, we have used two well-known cluster-validation measures [18]. The first is the standard classification rate. To compute the measure, for each cluster, we note the predominant shape class. Those graphs assigned to the cluster which do not belong to the predominant shape class are deemed to be misclassified. The classification rate is a fraction of graphs belonging to the predominant cluster shape classes divided by the total number of graphs. This measure exhibits a well-known bias toward a large number of classes. To overcome this, we also used the Rand index. The Rand index is defined as $R_I = \frac{A}{A+B}$. Here, A is the number of "agreements" that is the number of pairs of graphs that belong to the same class and that are assigned to the same cluster and B is the number of "disagreements," that is, the number of pairs of graphs that belong to different shape classes and that are assigned to different clusters. The index is hence the fraction of graphs of a particular shape class that are closer to an graph of the same class than to one of another class.

Fig. 10 plots the proportion of shapes correctly classified as the number of shape classes is increased. The mixture of tree unions appears as the curve marked with "boxes" and clearly outperforms the alternatives by a substantial margin, irrespective of which pairwise clustering method or which distance measure is used. The margin of improvement is greatest for large numbers of shape-classes. Fig. 11 compares the value of the Rand index obtained with the mixture of tree unions method and the values obtained with the alternative distance measures and clustering algorithms, as a function of the number of shape-classes. Again, the result of using the mixture of tree-unions is marked with "boxes." It is interesting to note that for our method and the spectral clustering method of Robles-Kelly and Hancock [30] the value of the Rand index increases with the number of shape classes present. This is due to the fact that the more shape-classes that are present, the less the measure punishes overfitting. With a small number of classes, the Perona-Freeman algorithm separates the graphs into a smaller number of clusters and, hence, produces the least overfitting. However, as we increase the number of shape classes, the mixture of tree unions method produces better results, outperforming both the Robles-Kelly and Hancock, and the Perona-Freeman pairwise clustering methods.

We now turn our attention to the results of applying PCA to the union trees, as described in Section 6. Fig. 12 displays the first two principal components of the sample-tree distribution for the embedding spaces extracted from six shape classes. In most cases, there appears to be a tightly packed central cluster with a few shapes scattered further away than the rest. This separation is linked to substantial variations in the structure of the shock trees. For example, in the shape-space formed by the class of pliers, the outlier is the only pair-of-pliers with the nose closed. In the case of the horse-class, the outliers appear to be the cart-horses while the inliers are the ponies.

7.3 Synthetic Data

To augment these real-world experiments, we have fitted the mixture of weighted tree unions to synthetically generated data. Our aim here has been to characterize the sensitivity of the algorithm to cluster merging. We have randomly generated a number of unweighted prototype trees and, from each tree, we have generated structurally perturbed copies. The procedure for generating the random trees was as follows: We commence with an empty tree (i.e., one with no nodes) and we iteratively add the required number of nodes. At each iteration, nodes are added as children of one of the existing nodes. The parents are randomly selected with uniform probability from among the existing nodes. The trees are perturbed by randomly adding the required number of nodes.

The sample of tree used in our study is controlled by increasing the number of prototypes, and increasing the degree of structural perturbation to which they are subjected. We tested the performance of the mixture of weighted tree unions, on samples generated from two, three, and four

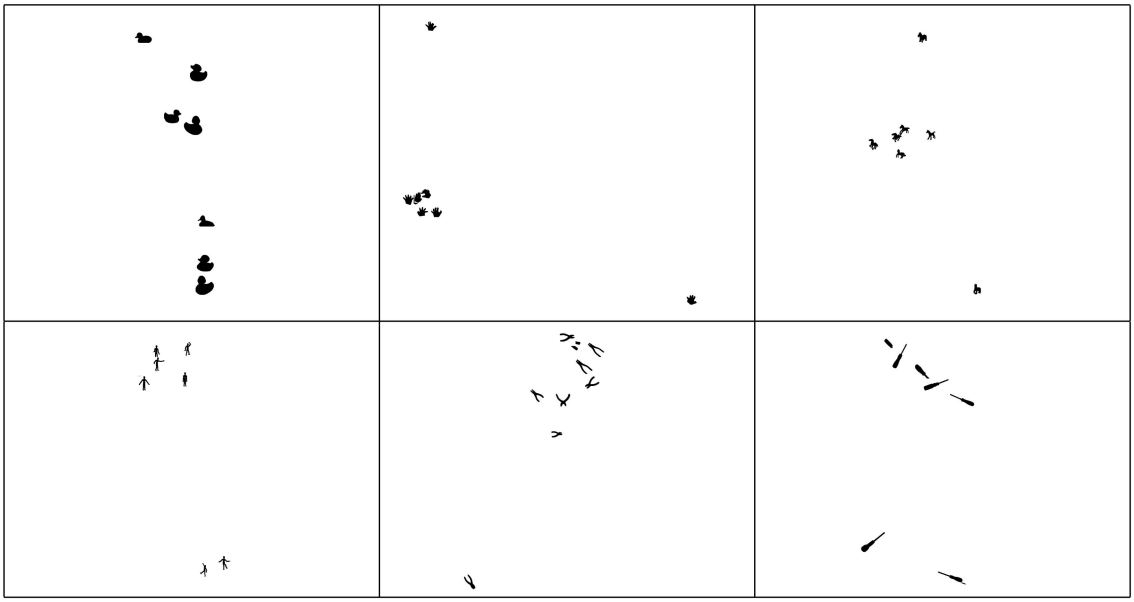

Fig. 12. Principal components analysis of the union embedding of the clusters.

prototypes of 10 nodes each. The amount of perturbation or noise is increased from an initial 10 percent to a maximum of 50 percent of the total number of nodes. Fig. 13 plots the fraction of pairs of trees that are correctly classified as belonging to the same or different clusters as the noise is increased. From this plot, it is clear that the method works well with compact and well-separated clusters. However, the algorithm undergoes a sudden drop in performance when the structural variability of the class reaches 40 percent of the total number of nodes of the prototypes. Furthermore, when more prototypes are used, then the distance between the clusters is smaller and, consequently the classes are harder to separate.

8 Conclusions

In this paper, we have presented an information theoretic framework for clustering trees and for learning a generative model of the variation in tree structure. The problem is posed as that of learning a mixture of tree unions. We demonstrate how the three sets of operations needed to learn the generative model, namely, node correspondence, tree merging, and node probability estimation, can each be couched in terms of minimizing a description length criterion. We provide variants of algorithm that can be applied to samples of both weighted and unweighted trees. Moreover, we illustrate the relationship between classical tree edit distance, and the node entropy in our model. The method is illustrated on the problem of learning shape-classes from sets of shock trees.

There are clearly a number of ways in which this work described in this paper may be extended. First, we have concentrated on trees, and there is scope for generalizing the method to graphs. Second, the method only accommodates node probabilities, and an important priority is to incorporate structures with weighted edges and to allow for edge-probabilities. Third, the optimization process is extremely simplistic, and prone to convergence to local optima. Hence, there is a need to investigate the use of more sophisticated methods such as mean field annealing or evolutionary search.

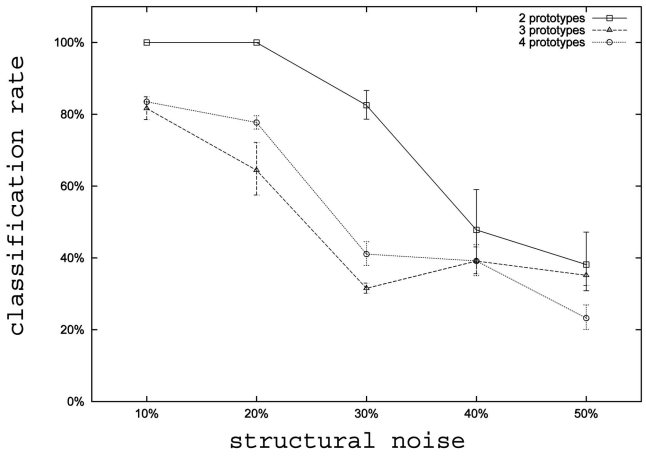

Fig. 13. Percentage of correct classifications under increasing structural noise.

References

[1] R. Baxter and J. Olivier, "MDL and MML: Similarities and Differences (Introduction to Minimum Encoding Inference—Part III)," Technical Report 207, Dept. of Computer Science, Monash Univ., 1994.

[2] H. Bunke et al., "Graph Clustering Using the Weighted Minimum Common Supergraph," *Graph Based Representations in Pattern Recognition*, pp. 235-246, 2003.

[3] C.J.K. Chow and C.N. Liu, "Approximating Discrete Probability Distributions with Dependence Trees," *IEEE Trans. Information Theory*, vol. 14, no. 3, pp. 462-467, 1968.

[4] T.F. Cootes, C.J. Taylor, and D.H. Cooper, "Active Shape Models—Their Training and Application," *Computer Vision and Image Understanding*, vol. 61, pp. 38-59, 1995.

[5] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *J. Royal Statistical Soc., Series B*, vol. 39, pp. 1-38, 1977.

[6] M.A. Eshera and K.-S. Fu, "An Image Understanding System Using Attributed Symbolic Representation And Inexact Graph-Matching," *IEEE Trans. Pattern Recognition and Machine Intelligence*, vol. 8, pp. 604-618, 1986.

[7] M.A.T. Figueiredo and A.K. Jain, "Unsupervised Learning of Finite Mixture Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 3, pp. 381-396, Mar. 2002.
[8] D. Fisher, "Knowledge Acquisition via Incremental Conceptual Clustering," *Machine Learning,* vol. 2, no. 2, pp. 139-172, 1987.
[9] N. Friedman, "Learning Bayesian Networks in the Presence of Missing Variables," *Proc. Int'l Conf. Machine Learning,* pp. 125-133, 1997.
[10] N. Friedman and D. Koller, "Being Bayesian about Network Structure," *Machine Learning,* vol. 50, nos. 1-2, pp. 95-125, 2003.
[11] L. Getoor et al., "Learning Probabilistic Models of Relational Structure," *Proc. Int'l Conf. Machine Learning,* pp. 170-177, 2001.
[12] P. Grünwald, "A Minimum Description Length Approach to Grammar Inference," *Proc. Symbolic, Connectionist, and Statistical Approaches to Learning for Natural Language Processing,* pp. 203-216, 1996.
[13] P. Grünwald, "Minimum Description Length Tutorial," *Advances in Minimum Description Length: Theory and Applications,* Apr. 2005.
[14] D. Heckerman, D. Geiger, and D.M. Chickering, "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data," *Machine Learning,* vol. 20, no. 3, pp. 197-243, 1995.
[15] M. Hagenbuchner, A. Sperduti, and A.C. Tsoi, "A Self-Organizing Map for Adaptive Processing of Structured Data," *IEEE Trans. Neural Networks,* vol. 14, pp. 491-505, 2003.
[16] T. Horváth, S. Worbel, and U. Bohnebeck, "Relational Instance-Based Learning with Lists and Terms," *Machine Learning,* vol. 43, pp. 53-80, 2001.
[17] S. Ioffe and D.A. Forsyth, "Human Tracking with Mixtures of Trees," *Proc. Int'l Conf. Computer Vision,* vol. I, pp. 690-695, 2001.
[18] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data.* Prentice Hall, 1988.
[19] B.J. Jain and F. Wysotzki, "Central Clustering of Attributed Graphs," *Machine Learning,* vol. 56, pp. 169-207, 2004.
[20] D.G. Kendall, "A Survey of Statistical Theory of Shape (with Discussion)," *Statistical Science,* vol. 4, no. 2, pp. 87-120, 1989.
[21] B.B. Kimia, A.R. Tannenbaum, and S.W. Zucker, "Shapes, Shocks, and Deformations I," *Int'l J. Computer Vision,* vol. 15, pp. 189-224, 1995.
[22] A.D. Lanterman, "Schwarz, Wallace and Rissanen: Intertwining Themes in Theories of Model Selection," *Int'l Statistical Rev.,* vol. 69, no. 2, pp. 185-212, 2001.
[23] T. Liu and D. Geiger, "Approximate Tree Matching and Shape Similarity," *Proc. Int'l Conf. Computer Vision,* pp. 456-462, 1999.
[24] M.A. Lozano and F. Escolano, "ACM Attributed Graph Clustering for Learning Classes of Images," *Graph Based Representations in Pattern Recognition,* pp. 247-258, 2003.
[25] M. Meilă, "Learning with Mixtures of Trees," PhD thesis, Mass. Inst. of Technology, 1999.
[26] R.L. Ogniewicz, "A Multiscale Mat from Voronoi Diagrams: The Skeleton-Space and Its Application to Shape Description and Decomposition," *Aspects of Visual Form Processing,* pp. 430-439, 1994.
[27] R. Otter, "The Number of Trees," *Ann. Math.,* vol. 49, pp. 583-599, 1948.
[28] P. Perona and W.T. Freeman, "A Factorization Approach to Grouping," *Proc. European Conf. Computer Vision,* vol. 1, pp. 655-670, 1998.
[29] J. Rissanen, "Modelling by Shortest Data Description," *Automatica,* vol. 14, pp. 465-471, 1978.
[30] A. Robles-Kelly and E.R. Hancock, "A Probabilistic Spectral Framework for Segmentation and Grouping," *Pattern Recognition,* vol. 37, pp. 1387-1406, 2004.
[31] A. Sanfeliu and K.S. Fu, "A Distance Measure Between Attributed Relational Graphs for Pattern Recognition," *Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics,* vol. 13, pp. 353-362, 1983.
[32] A. Shokoufandeh, S.J. Dickinson, K. Siddiqi, and S.W. Zucker, "Indexing Using a Spectral Encoding of Topological Structure," *Computer Vision and Pattern Recognition,* vol. 2, pp. 491-497, 1999.
[33] K. Siddiqi et al., "Shock Graphs And Shape Matching," *Int'l J. Computer Vision,* vol. 35, pp. 13-32, 1999.
[34] E. Klassen, A. Srivastava, W. Mio, and S.H. Joshi, "Analysis of Planar Shapes Using Geodesic Paths on Shape Spaces," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 3, pp. 372-383, Mar. 2004.
[35] A. Torsello and E.R. Hancock, "A Skeletal Measure of 2D Shape Similarity," *Computer Vision and Image Understanding,* vol. 95, no. 1, pp. 1-29, 2004.
[36] A. Torsello and E.R. Hancock, "Correcting Curvature-Density Effects in the Hamilton-Jacobi Skeleton," *IEEE Trans. Image Processing,* vol. 15, no. 4, pp. 877-891, 2006.
[37] A. Torsello and E.R. Hancock, "Efficiently Computing Weighted Tree Edit Distance Using Relaxation Labeling," *Energy Minimization Methods in Computer Vision and Pattern Recognition,* 2001.
[38] A. Torsello, D. Hidovic-Rowe, M. Pelillo, "Polynomial-Time Metrics for Attributed Trees," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 27, no. 7, pp. 1087-1099, July 2005.
[39] C. Wallace and D. Burton, "An Information Measure for Classification," *The Computer J. ,* vol. 11, no. 2, pp. 195-209, 1968.
[40] J.H.M. Wedderburn, "The Functional Equation $g(x^2) = 2ax + [g(x)]^2$," *Ann. Math.,* vol. 24, pp. 121-140, 1922-23.
[41] S.C. Zhu and A.L. Yuille, "FORMS: A Flexible Object Recognition and Modelling System," *Int'l J. Computer Vision,* vol. 20, no. 3, pp. 187-212, 1996.

Andrea Torsello received the "Laurea" degree with honors in computer science from Ca' Foscari University of Venice, Italy, in 1997. In 2004, he received the PhD degree in computer science at the University of York, United Kingdom. Currently, he is an assistant professor at Ca' Foscari University of Venice, Italy. His research interests are in the area of computer vision and pattern recognition, in particular, the application of stochastic and structural approaches to shape analysis. He is currently coediting a special issue of *Pattern Recognition* on "similarity-based pattern recognition."

Edwin R. Hancock studied physics as an undergraduate at the University of Durham and graduated with honors in 1977. He remained at Durham to complete the PhD degree in the area of high-energy physics in 1981. Following this, he worked for 10 years as a researcher in the fields of high-energy nuclear physics and pattern recognition at the Rutherford-Appleton Laboratory (now the Central Research Laboratory of the Research Councils). During this period, he also held adjunct teaching posts at the University of Surrey and the Open University. In 1991, he moved to the University of York as a lecturer in the Department of Computer Science. He was promoted to senior lecturer in 1997 and to reader in 1998. In 1998, he was appointed to a Chair in Computer Vision. Professor Hancock now leads a group of some 15 faculty, research staff, and PhD students working in the areas of computer vision and pattern recognition. His main research interests are in the use of optimization and probabilistic methods for high and intermediate level vision. He is also interested in the methodology of structural and statistical pattern recognition. He is currently working on graph-matching, shape-from-X, image databases, and statistical learning theory. His work has found applications in areas such as radar terrain analysis, seismic section analysis, remote sensing, and medical imaging. He has published more than 90 journal papers and 350 refereed conference publications. He was awarded the Pattern Recognition Society medal in 1991 and an outstanding paper award in 1997 by the *Journal of Pattern Recognition*. In 1998, he became a fellow of the International Association for Pattern Recognition. Professor Hancock has been a member of the editorial boards of the journals *IEEE Transactions on Pattern Analysis and Machine Intelligence*, and *Pattern Recognition*. He has also been a guest editor for special editions of the journals *Image and Vision Computing* and *Pattern Recognition*. He has been on the program committees for numerous national and international meetings. In 1997, with Marcello Pelillo, he established a new series of international meetings on energy minimisation methods in computer vision and pattern recognition.