Proceedings, COLT'89, Santa Cruz, Ca. Aug, 1989.
pp. 230-244.

TECHNICAL REPORT
R-132
June, 1989

LEARNING STRUCTURE FROM DATA: A SURVEY [(*)]

Judea Pearl
Computer Science Dept.
University of California
Los Angeles, CA. 90024-1596

Rina Dechter
Dept. of Computer Sciences
Technion, I. I. T.
Haifa, Israel

ABSTRACT

This paper summarizes several investigations into the prospects of identifying meaningful structures in empirical data. Starting with an early work on identifying probabilistic trees, we extend the method to polytrees (directed trees with arbitrary edge orientation) and show that, under certain conditions, the skeleton of the polytree as well as the orientation of some of the arrows, are identifiable. We next address the problem of identifying probabilistic trees in which some of the nodes are unobservable. It is shown that such trees can be effectively identified in cases where all variables are either bi-valued or normal, and where all correlation coefficients are known precisely. Finally, it is shown that an effective procedure exists for determining whether a given categorical relation is decomposable into a tree of binary relations and, if the answer is positive, identifying the topology of such a tree. Guided by these results, we then propose a general framework whereby the notion of *identifiability* is given a precise formal definition, similar to that of learnability.

1. INTRODUCTION

Discovering meaningful structures in empirical data has long been regarded as the hallmark of scientific activity. Formally, the task of finding "meaningful structures" can be stated as that of finding computationally attractive descriptions of the data whenever such descriptions exist. Invariably, the existence of useful descriptions rests on whether the dependencies among the data items are decomposable into a small number of more basic interactions. Given that the data was generated from a model where the dependencies are both sparse and local, one seeks a useful representation for these dependencies. A classical example would be to find a Markov model with the least number of states that accounts for observed dependencies among contingent symbols in a sequence. In more elaborate settings the dependencies can form a graph (as in the analysis of Markov fields) or a hypergraph (as in relational databases), and the task is to find the topology of these structures. Structure learning includes such tasks as finding effective representations for probability distributions, finding economical decompositions of database schema, or simplifying expressions of learned Boolean concepts to render subsequent processing tractable.

* This work was supported in part by National Science Foundation Grants #IRI-86-10155 and #IRI-8815522, and Naval Research Laboratory Grant #N00014-89-J-2007.

Despite the generality and importance of the task at hand, very few formal results have been established. In the economics literature the topic has been discussed under the heading of "latent structure analysis" [Lazarsfeld 1966; Glymour et. al. 1987], while in the pattern-recognition literature it became known as "unsupervised learning" [Duda and Hart 1973]. However, with the exception of the work of Chow and Liu [1968] the tasks were confined to learning very simple structures, such as those governing probability mixtures and hidden Markov models [Laird 1988].

This paper surveys several extensions of the Chow and Liu result (Section 2). These include identifying polytrees (Section 3), identifying trees with hidden variables (Section 4), and identifying tree decompositions of categorical relations (Section 5). A general formal framework for structure identification tasks and a comparison to Valiant's [1984] learning model are provided in Section 6.

2. IDENTIFYING TREES

Definition: A distribution $P^t(\mathbf{x})$ is said to be *tree-dependent* relative to the tree t; if it can be written as a product of pair-wise conditional probability distributions,

$$P^t(\mathbf{x}) = \prod_{i=1}^{n} P(x_i \mid x_{j(i)}) , \tag{1}$$

where $X_{j(i)}$ is the variable designated as the parent of X_i in some orientation of the tree.

The root X_1 can be chosen arbitrarily, and having no parents, it is characterized by the prior probability $P(x_1 \mid x_0) = P(x_1)$. For example, the distribution corresponding to the tree oriented as in Figure 1 has the product form

$$P^t(\mathbf{x}) = P(x_1) P(x_2 \mid x_1) \, P(x_3 \mid x_2) \, P(x_4 \mid x_2) \, P(x_5 \mid x_2) \, P(x_6 \mid x_5) \, .$$

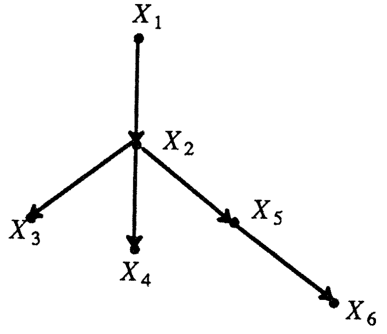


Figure 1.

Thus, the number of parameters needed for specifying a tree-dependent distribution is $(r-1)[r(n-1)+1]$, where n is the number of variables and r their arity. Tree-dependent distributions are unique in that they facilitate linear time query processing, as well as local unsupervised computations [Pearl 1988].

Chow and Liu asked the following question: Given a distribution P, what is the tree-dependent distribution P^t that best approximates P, in the sense of minimizing the Kullback-Liebler measure

$$D(P, P') = \sum_{\mathbf{x}} P(\mathbf{x}) \log \frac{P(\mathbf{x})}{P'(\mathbf{x})} \quad ? \tag{2}$$

This measure is nonnegative and attains the value 0 if and only if P' coincides with P.

The minimization task can be performed in two surprisingly simple steps. First, we fix the structure of some tree t and ask what conditional probabilities $P^t(x_i \mid x_j)$ would render P^t the best approximation of P. We call this best approximation the *projection* of P on t, P_P^t. Second, we vary the structure of t over all possible spanning trees, and among all the projections of P on these spanning trees we seek the one that is closest to P.

Theorem 1: *The projection of P on t is characterized by the equality*

$$P_P^t(x_i \mid x_{j(i)}) = P(x_i \mid x_{j(i)}) \, . \tag{3}$$

In other words, by forcing the conditional probabilities along the branches of the tree t to coincide with those computed from P, we get the best t-dependent approximation to P. (see Pearl [1988], Appendix 8-A, for proof).

Theorem 2 [Chow and Liu 1968]: *The distance measure of Eq. (2) is minimized by projecting P on any maximum weight spanning tree (MWST), where the weight on the branch (X_i, X_j) is defined by the mutual information measure*

$$I(X_i, X_j) = \sum_{x_i, x_j} P(x_i, x_j) \log \frac{P(x_i, x_j)}{P(x_i)\, P(x_j)} \geq 0 \, . \tag{4}$$

Corollary 1: *If the underlying distribution P is itself tree-dependent, then its projection on every MWST must coincide with P.*

The merits of the MWST algorithm are several. First, it uses only second-order statistics, which are easily and reliably measured from sample data and are economical to store. The tree is developed in $O(n^2)$ steps using only weight comparisons, thereby avoiding expensive tests for conditional independence. Finally, it can be shown that if the branch weights are computed from sampled data, then P_P^t will be a maximum likelihood estimate of P. The consequence of this is that if the underlying distribution is indeed one of tree-dependence, the approximating probability constructed by the MWST algorithm converges with probability 1 to the true underlying distribution [Chow and Wagner 1973].

These advantages are unique to tree structures and do not apply to general multiply connected graphs. Even if we could afford to enumerate all graphs in a given class, the task of finding the projection of P on a product form that reflects any of these graphs might involve enormous computations, unless that graph could be embedded in a simple hyper-tree.

3. IDENTIFYING POLYTREES

A *polytree* is a directed tree with unrestricted edge orientation, thus allowing nodes to have multiple parents (see Figure 2). A polytree, though it allows us to describe higher-order interactions, enjoys many of the computational advantages of simple trees. In particular, it supports local computations, and its structure can sometimes be identified by second-order distributions using an MWST algorithm similar to that of Chow and Liu.

Assume we are given a distribution $P(\mathbf{x})$ of n discrete-value variables, and we are told that $P(\mathbf{x})$ can be represented by some unknown polytree F_0, i.e., $P(\mathbf{x})$ has the form

$$P(\mathbf{x}) = \prod_{i=1}^{N} P(x_i \mid x_{j_1(i)}, x_{j_2(i)}, \ldots, x_{j_m(i)}) \,, \tag{5}$$

where $\{X_{j_1(i)}, X_{j_2(i)}, \ldots, X_{j_m(i)}\}$ is the (possibly empty) set of direct parents of variable X_i in F_0, and the parents of each variable are mutually independent, i.e.,

$$P(x_{j_1(i)}, x_{j_2(i)}, \ldots, x_{j_m(i)}) = \prod_{k=1}^{m} P(x_{j_k(i)}) \quad \text{for all } i \,. \tag{6}$$

We seek to recover the structure of F_0, i.e., the set of branches and hopefully their directionality, by examining the properties of $P(\mathbf{x})$. That the structure of F_0 may not always be recoverable uniquely is apparent from examining the dependencies induced by the three possible types of adjacent triplets allowed in polytrees:

1. $X \rightarrow Y \rightarrow Z$

2. $X \leftarrow Y \rightarrow Z$

3. $X \rightarrow Y \leftarrow Z$

Since type 1 and type 2 represent the same dependencies among X, Y and Z, they are indistinguishable. Type 3, however, can be uniquely identified, since X and Z are marginally independent and all other pairs are dependent. Thus, while the *skeletons* (the graphs stripped of arrows) of these three triplets are identical, the directionality of the arrows is partially identifiable.

Definition: A distribution $P(\mathbf{x})$ is said to be ***nondegenerate*** if there exists a connected directed acyclic graph (DAG) that displays all the dependencies and independencies embedded in P.

Theorem 3: *If a nondegenerate $P(\mathbf{x})$ can be represented by a polytree F_0 (as in (5) and (6)), then the MWST algorithm of Chow and Liu unambiguously recovers the skeleton of F_0.*

Definition: A *causal basin* of a node A in a DAG is a set of nodes consisting of A, the direct parents of A, all the descendants of A and all of the direct parents of those descendants.

Theorem 4 [Rebane and Pearl 1987]: *If a nondegenerate $P(\mathbf{x})$ is representable by a polytree F_0, then the directionality of a branch can be recovered if and only if it is contained within the causal basin of some multi-parent node in F_0.*

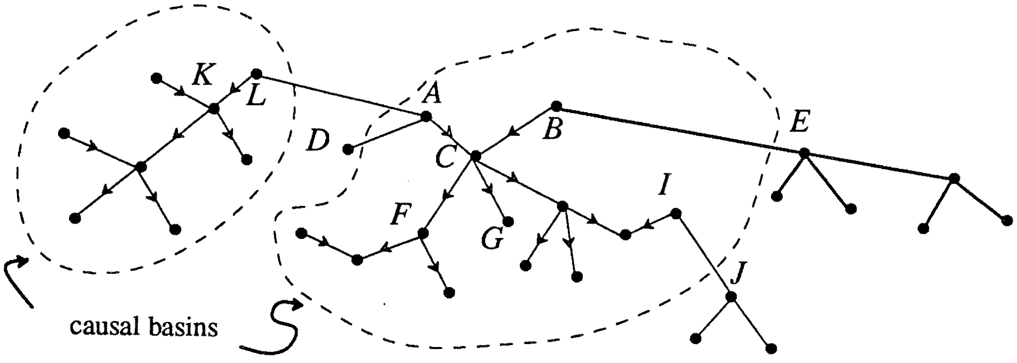


Figure 2.

Figure 2 depicts a polytree with two causal basins rooted as nodes K and C. The arrows in each causal basin can be determined uniquely, while the directions of branches outside the basins are undetermined.

Under conditions of degeneracy, $P(\mathbf{x})$ can be represented by several polytrees, each having a different skeleton or a different branch orientation, or both. In general, while we are not guaranteed that every skeleton produced by the MWST algorithm will permit a faithful representation of P, we are guaranteed that at least one of these skeletons will do so. This is because if P can be represented by a set of k distinct skeletons $\mathbf{T} = (T_1,...,T_k)$, then each of these skeletons (and perhaps others) must have maximum weight.

4. IDENTIFYING TREES WITH HIDDEN VARIABLES

The identification methods described in sections 2 and 3 assume that the interactions among the observed variables can be decomposed into the desired structures. It is often the case that decomposition is hindered when some variables remain unobserved or *hidden,* in which case the task is to postulate the existence of such variables and identify their connections to the observables. The task of learning mixture distributions [Duda and Hart 1973] has this flavor; we postulate the existence of a hidden variable W such that observed distribution $P(\mathbf{x})$ can be written as sums of products

$$P(x_1, x_2, ..., x_n) = \sum_{j=1}^{\lambda} \prod_{i=1} P(x_i \mid w_j) P(w_j),$$

and attempt to find reasonable estimates of λ, $P(w_j)$ and $P(x_i \mid w_j)$. Graphically, such decomposition corresponds to a *star* network, since the visible variables (the X_i's) are connected star-like to one central hidden variable W.

Lazarsfeld [1966] considered star-decomposable distributions where the visible variables are bi-valued and the hidden variable W is permitted to range over λ values, $\lambda>2$. The identification of $P(x_i \mid w_j)$ requires the solution of $\lambda n+\lambda-1$ nonlinear equations to find the values of the $\lambda n+\lambda-1$ independent parameters. Letting $\lambda = 2^n - 1$ yields a trivial, unconstrained solution, where each value of W corresponds to one entry of the joint distribution function.

To maintain uniformity in the tree, we can postulate several hidden binary variables but insist that they form a treelike structure, i.e., each triplet of leaves forms a star, but the central variables may differ from triplet to triplet.

Definition: We shall say that a distribution $P(x_1, x_2,..., x_n)$ is *tree-decomposable* if it is the marginal probability of a tree-dependent distribution

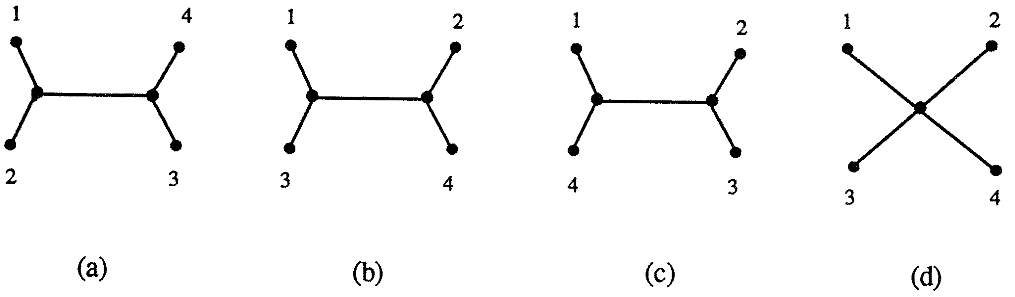
$$P_T(x_1, x_2,..., x_n, w_1, w_2,..., w_m) \qquad m \le n-2,$$

such that $W_1, W_2,..., W_m$ correspond to the internal nodes of a tree T and $X_1, X_2,..., X_n$ correspond to its leaves. We say that P_T is a *tree-extension* of P, and P is the *leaf-marginal* of P_T. P_T is said to be a *minimal* tree-extension of P if no variable or link can be deleted from the tree T without P_T ceasing to be a tree-extension of P. In the example of Figure 1, $P(x_1, x_3, x_4, x_6)$ is tree-decomposable, with $P(x_1, x_2, x_3, x_4, x_5, x_6)$ as its tree-extension.

Given a tree-decomposable distribution $P(x_1 ,..., x_n)$, we ask if it is possible to find any minimal extension $P_T(x_1 ,..., x_n, w_1 ,..., w_m)$ of P.

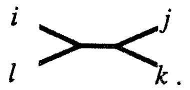
Theorem 5: *If all variables are either bi-valued or normal, then (1) the minimal tree-extension P_T is unique up to renaming of the variables or their values, (2) P_T can be recovered from P using $n \log n$ computations, and (3) the structure of T is uniquely determined by the second-order probabilities of P.*

The method of constructing P_T [Pearl and Tarsi 1986] is based on the observation that every four leaves in a tree are interconnected in one of four distinct topologies, depending on which pairs can be connected via non-intersecting paths (see Figure 3). Moreover, each of the four topologies is characterized by a unique equality among the leaves' correlation coefficients, which can be determined empirically. For example, the topology



(a) (b) (c) (d)

Figure 3.



is characterized by the equation $\rho_{ij}\rho_{kl} = \rho_{ik}\rho_{jl}$.

This method of constructing a tree-extension has two major shortcomings: it requires precise knowledge of the correlation coefficients, and it works only when the underlying model is tree-decomposable. In practice, we often have only sample estimates of the correlation coefficients; therefore, it is important to find a tree-structured *approximation* for a distribution that is not tree-decomposable. Unfortunately, no method is known which provides a guarantee on the quality of such an approximation.

5. IDENTIFYING TREE STRUCTURES IN CATEGORICAL RELATIONS

Let ρ denote an n-ary relation over the set of attributes $U = \{X_1, \ldots, X_n\}$, i.e., a subset of the Cartesean product $Dom(X_1) \times, \ldots, \times Dom(X_n)$, where $Dom(X_i)$ is the set of values of attribute X_i. Let ρ_S denote the projection of ρ on a subset S of attributes, namely, ρ_S is obtained from ρ by striking out columns corresponding to attributes $U-S$ and removing duplicate tuples in what remains. Given an attribute A and an element of its domain a, the **restriction of** ρ to $A=a$, denoted by $\rho^{r(A=a)}$, is the n-ary relation containing all n-tuples in ρ having value a for A. Similarly, the restriction of ρ to a subtuple $t = (X_{i1} = x_{i1}, \ldots, X_{ir} = x_{ir})$, denoted by $\rho^{r(t)}$ is all the n-tuples of ρ that match t for the corresponding attributes.

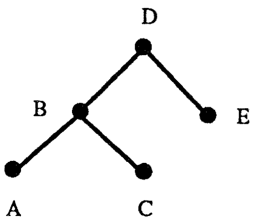
Definition: Let S_1, S_2, S_3 be subsets of U. S_1 and S_2 are *conditionally independent* given S_3, denoted by $<S_1 \mid S_3 \mid S_2>$, if for every combination of values for attributes in S_3, denoted by $S_3 = s_3$, we have

$$L^{r(S_3=s_3)} = (L_{S_1S_3})^{r(S_3=s_3)} \times (L_{S_2S_3})^{r(S_3=s_3)},$$

where

$$L = \rho_{S_1S_2S_3}.$$

Conditional independence parallels the notion of Embedded-Multi-Valued-Dependencies (EMVDs) [Maier 1983]. That is, if $<S_1 \mid S_3 \mid S_2>$, holds in a relation ρ, and $S_1 = U - S_3S_2$, then ρ can be decomposed losslessly into the database scheme S_1S_3 and S_2S_3. A relation is said to be *tree-decomposable* if it can be decomposed losslessly into a tree of binary relations (also known as *constraint-tree)*. For example, the constraint-tree shown in Figure 4 (together with the binary relations on its arcs) represents a lossless decomposition of the relation given in Figure 5. This can be verified by enumerating the 5-tuples that satisfy all the binary relations in Figure 4, and comparing to those listed in the table of Figure 5. The computational advantages of constraint-tree representations are that they permit queries of enumeration, extension and entailment to be answered in polynomial time.



D	B
1	0
1	1
0	1

D	E
1	1
1	0
0	1

A	B
0	0
0	1

B	C
0	1
1	1
1	0

Figure 4.

A	B	C	D	E
0	0	1	1	1
0	0	1	1	0
0	1	1	1	1
0	1	1	1	0
0	1	1	0	1
0	1	0	1	1
0	1	0	1	0
0	1	0	0	1

Figure 5.

A constraint graph (or network), representing a relation, explicates some of its conditional independencies. In a graph, S_2 is said to **separate** S_1 from S_3 if by removing the nodes in S_2, nodes in S_1 are disconnected from nodes in S_3. Every **separation** in the constraint graph corresponds to a conditional independence, i.e., if a subset S_1 separates (in the constraint graph) subset S_2 from S_3 then $<S_2 \mid S_1 \mid S_3>$. However, there may be conditional independencies in the relation which are not manifested in the constraint graph. For a detailed discussion of conditional-independencies and their graphical representation see [Pearl and Paz 1986; Pearl 1988]. Constraint networks are discussed in [Montanari 1974] and [Dechter and Pearl 1987].

The possibility of extending Chow and Liu's result from probabilities to categorical relations rests on the following questions. Suppose that a relation ρ is associated with a uniform probability distribution, U_ρ, which accords equal non-zero probabilities to all tuples in the relation and only to those tuples. Does the existence of an exact tree-dependent distribution for this uniform distribution imply that the relation is tree-decomposable? Moreover, does a tree-decomposable relation necessarily have a tree-dependent uniform probability distribution? Are they decomposable by the same tree? And if so, can we use the MWST method to find an optimal tree-decomposable approximation to a given relation? These questions were analyzed in [Dechter 1987]; the answers were shown to be in the affirmative for exact decompositions but remain unsettled for approximations.

Theorem 6: *If a probability distribution P is tree-dependent along a tree T, then the relation defined by the possible tuples of P ($P > 0$) has a lossless decomposition by the same tree.*

Theorem 7: *For any relation ρ, if ρ can be decomposed losslessly into some tree T, then its associated uniform distribution, U_ρ, is tree-dependent along T.*

Thus, applying the MWST to U_ρ, we obtain the following theorem.

Theorem 8: *Let* $n(x_i)$ be the number of n-tuples in ρ for which $X_i = x_i$, and let $n(x_i, x_j)$ be the number of n-tuples in ρ for which $X_i = x_i$ and $X_j = x_j$. The MWST algorithm using the arc-weights:

$$m(X_i, X_j) = \frac{1}{|\rho|} \sum_{(x_i, x_j) \in \rho_{X_i X_j}} n(x_i, x_j) \log \frac{n(x_i, x_j)}{n(x_i) n(x_j)} \tag{7}$$

is guaranteed to produce a tree-decomposition to ρ if such a decomposition exists.

The following algorithm takes a relation ρ and returns a set of tree-structured binary relations, which are guaranteed to be lossless if the relation is tree-decomposable.

Tree-generation-algorithm

a. Compute the basic quantities: $n(x_i)$ and $n(x_i, x_j)$.

b. For every two attributes X_i , X_j compute the weights $m(X_i, X_j)$ given in Eq. (7).

c. Find a **maximum weight spanning tree** of the complete graph w.r.t. the above arc-weights.

d. For each pair of attributes that corresponds to an arc in the selected tree find the associated relation by projecting the global relation on it.

The complexity of the algorithm is $O((l + \log n) n^2)$, where n is the number of attributes and l is the size of the relation. To verify that the generated tree represents the input relation, we compute (in linear time) the number of n-tuples represented by the tree-decomposition and compare it to the size of the given relation. If the two numbers are equal, the database losslessly represents the relation. Otherwise, we know that no tree representation exists.

When no tree-decomposition exists, the algorithm produces a decomposition which is not lossless but can be regarded as an approximation to ρ. The decomposition algorithm, when applied to U_ρ, finds T such that the projection of U_ρ on T is the best approximation to U_ρ, with respect to the proximity measure of Eq. (2). However, it is not clear how this proximity measure translates into meaningful merit criteria for relations, such as the number of tuples in the approximating relation. Experiments show [Dechter 1987] that the MWST provides a very tight approximation whenever the relation ρ is expressible as a network of binary constraints, tighter than any other known method of tree-decomposition. We also know that the tree produced by the MWST method is the smallest possible whenever the relation defined by joining all pair-wise projections of ρ (so-called the *minimal network of* ρ) is tree-decomposable. However, whether this tree is the most specific one in the general case (i.e., whether it does not properly contain another tree-decomposable superset of ρ), remains an open problem. A non-numeric method of identifying tree-decompositions is developed in [Dechter, Meiri & Pearl 1989].

6. IDENTIFIABILITY VS. LEARNABILITY

The main difference between the problems described in this paper and those addressed by Valiant's model of learning is that in the latter we are given the concept class C and our concern is to infer which individual member of C is responsible for the observed instances. By contrast, in structure learning we are not given the concept class C. Rather, our main concern is to decide whether a fully observed concept, taken from some broad class C' (e.g., all categorical relations), is also a member of a narrower class C of concepts, one that possesses some desirable syntactical features (e.g., constraint-trees). Thus, the task is not to identify the semantic extension of a concept (this is assumed to be directly observed), but to identify its syntactical description.

It turns out that casting a concept in a convenient description requires more than clever syntactic manipulations. The feasibility of finding such a description rests heavily on the nature of the dependencies among the components of the concept (e.g., the attributes or the predicates), and these dependencies must be uncovered from the semantic extension of the concept. Moreover, finding a desirable description to a given concept, even when it is feasible and even when the concept is of small size, might require insurmountable computation; a problem not normally addressed in traditional models of learning.

To cast these considerations in a general formal framework, we define the notion of *Identifiability*, and contrast it with that of *Learnability*.

Definition: (Identifiability) A class of concepts C is said to be identifiable relative to a background class C', iff:

(1) (Recognition) For every set I of instances, representing a concept in C', there is an algorithm A, polynomial in $|I|$, that determines if I is a member of C, and

(2) (Isolation) If the answer to (1) is positive, A finds one member of C that matches I.

Definition: (Strong Identifiability) Same as (1) and (2) above, with the addition of:

(3) (Specificity) If the answer to (1) is negative, the algorithm finds a *minimal* concept c_0 in C that contains I, i.e., $I \subset c_0$, and there is no $c \in C$ such that $I \subset c \subset c_0$.

By convention, a class in which the recognition or isolation tasks are NP-hard will be defined as non-identifiable. Moreover, the concept size $|I|$ should measure the overall code length used in the description of I, not merely the number of instances in I.

EXAMPLES

1. **(Constraint networks)** Let C' be the set of all relations on n attributes. The class $C_N (\subset C')$ of relations expressible by *binary networks* of constraints is not identifiable. Although we have algorithms for meeting requirement (3) (and hence (2)) of constructing the most specific network (so called minimal network) for any given relation I, we do not have an effective way of testing whether the minimal network represents the relation I exactly, or a superset thereof. Even generating a single instance of the minimal network might be an NP-complete problem, if the attributes are non-binary.

2. If the background class C' is known in advance to consist of only relations that are expressible by binary constraint networks (i.e., $C' = C_N$), then it is possible to identify such a network in time linear in the number of instances. Thus, C_N is strongly identifiable relative to itself. Under this condition it is also learnable, being a variant of $k-CNF$.

3. In general, if we set $C' = C$, then, if C is learnable it must also be strongly identifiable, because condition (1) is satisfied automatically, and the learnability requirement of zero error on negative examples is equivalent to (3). (Note that since the learner is entitled to observe the entire concept, the second learnability requirement of limited generalization errors plays no role in identifiability tasks.) However, there are concept classes that are identifiable but not learnable under the condition $C' = C$, a trivial example of which is the set of subsets having size $|I| = k$. A more significant example is described next.

4. **(Constraint Trees)** The class C_T $(= C')$ of *constraint-trees* is identifiable but not learnable. The MWST algorithm (discussed in Section 5) correctly identifies a tree that represents the instances. Yet, C_T is not learnable because it is not closed under intersection, i.e., there is no unique, most specific tree that captures every subset of instances drawn from a tree.

5. Section 5 shows that the class of constraint-trees is identifiable also relative to the background class of *all* relations. However, it is still an open question whether this class is strongly identifiable; we were not able to prove (or disprove) that, when the minimal network is not tree-decomposable, the MWST method still returns a minimal tree.

6. **(Constraint Chains)** The class C_C $(= C')$ of *constraint chains* is not learnable nor identifiable. The reason being that, since chains are not matroids, we do not have a greedy algorithm similar to the MWST for identifying the correct ordering of the variables.

7. **(Partial Orders)** Let C $(= C')$ be the class of all partial orders on n objects. C is known to be learnable (hence identifiable). However, if C $(= C')$ is the class of partial orders representable by a tree then it is not learnable but strongly identifiable (even if we

let C' extend to the class of all partial orders). (*)

8. **(Rooted Trees)** Let C' be all subsets of ordered triples (x, y, z) taken from a collection of n objects, and let each member c of C be the set of triplets (x, y, z) generated by a rooted tree with n leaves, such that (x, y, z) is in c iff the deepest common ancestor of x and y is also the deepest common ancestor of x and z. C is strongly identifiable in $O(n \ logn)$ time (see Pearl and Tarsi, 1985).

CONCLUSIONS

This paper summarizes several investigations into the prospects of identifying meaningful structures in empirical data. The central theme is to identify the topology of a tree of dependencies, in cases where the observed data possess such dependencies. Starting with an early work of Chow and Liu [1968] on identifying the best tree-dependent approximation to a given probability distribution, we extend the method to polytrees (directed trees with arbitrary edge orientation) and show that, under certain conditions, the skeleton of the polytree as well as the orientation of some of the arrows, are identifiable. We next address the problem of identifying probabilistic trees in which some of the nodes are unobservable. It is shown that such trees can be effectively identified in cases where all variables are either bi-valued or normal, and where all correlation coefficients are known precisely. Finding the best tree approximation to data that does not lend itself to precise tree representation (with hidden variables) remains an open problem. Finally, extending the task from probabilistic to categorical data, it is shown that an effective procedure exists for determining whether a given relation is decomposable into a tree of binary relations and, if the answer is positive, identifying the topology of such a tree. The procedure runs in time proportional to the size of the relation and can be used to provide an approximate, representation to a set of observed tuples, which is economical in both storage space and query processing time.

The task of finding a desirable description for a concept has been given a formal definition through the notion of identifiability, which is normally weaker (if $C' = C$) than that of learnability. Possibly due to their matroid properties, constraint-trees (as well as probabilistic trees) were found to be one of the very few useful structures which is identifiable yet not learnable.

Acknowledgements

We thank Othar Hansson, Itay Meiri and Amir Weinshtain for useful discussions on the notion of identifiability.

(*) This example is due to Othar Hansson.

References

C. K. Chow, & C.N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Trans. on Info. Theory,* IT-14:462-67 (1968)

C.K. Chow & T.J. Wagner. Consistency of an Estimate of Tree-dependent Probability Distributions. *IEEE Trans. on Info. Theory,* IT-19:369-71 (1973)

R. Dechter. Decomposing a Relation into a Tree of Binary Relations. In *Proceedings,* 6th Conf. on Princ. of Database Systems, San Diego, CA., pp. 185-189 (March 1987). To appear in *Journal of Computer and System Science,* Special Issue on the Theory of Relational Databases.

R. Dechter & J. Pearl. Network-Based Heuristics for Constraint-Satisfaction Problems. *Artificial Intelligence,* Vol. 34(1), pp. 1-38 December (1987)

R. Dechter, I. Meiri & J. Pearl. On the Identifiability of Binary Relations. UCLA Cognitive Systems Laboratory, *Technical Report, (R-140).* In preparation (1989)

R.O. Duda & P.E. Hart. *Pattern Recognition and Scene Analysis.* New York: Wiley (1973)

C. Glymour, R. Scheines, P. Spirtes, & K. Kelly. *Discovering Causal Structure.* New York: Academic Press (1987)

P.D. Laird. Efficient Unsupervised Learning. In *Proceedings,* 1988 Workshop on Computational Learning Theory, Haussler and Pitt (eds), San Mateo: Morgan Kaufmann, pp. 297-311 (1988)

P.F. Lazarsfeld, P.F. Latent Structure Analysis. In *Measurement and Prediction,* ed. S.A. Stouffer, L. Guttman, E.A. Suchman, P.F. Lazarsfeld, S.A. Star, and J.A. Claussen. New York: Wiley (1966)

D. Maier. *The Theory of Relational Databases,* Rockville, Maryland: Computer Science Press (1983)

U. Montanari. Networks of Constraints, Fundamental Properties and Applications to Picture Processing. *Information Science* Vol 7:95-132 (1974)

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference,* San Mateo: Morgan Kaufmann Publishers (1988)

J. Pearl & A. Paz. On the Logic of Representing Dependencies by Graphs. In *Proceedings,* 1986 Canadian AI Conference, Montreal, Canada, pp. 94-98 (1986)

J. Pearl & M. Tarsi. Structuring Causal Trees. *Journal of Complexity* Vol. 2(1):60-77 (1986)

G. Rebane & J. Pearl. The Recovery of Causal Poly-trees from Statistical Data. *Proceedings,* 3rd Workshop on Uncertainty in AI, Seattle, Washington, pp. 222-228 (1987)

L.G. Valiant. A Theory of the Learnable. *Communications of the ACM,* Vol. 27(11):1134-1142, November (1984)