Learning Probabilistic Models of Relational Structure

Lise Getoor GETOOR@CS.STANFORD.EDU
Computer Science Dept., Stanford University, Stanford, CA 94305

Nir Friedman NIR@CS.HUJI.AC.IL
School of Computer Sci. & Eng., Hebrew University, Jerusalem, 91904, Israel

Daphne Koller KOLLER@CS.STANFORD.EDU
Benjamin Taskar BTASKAR@CS.STANFORD.EDU
Computer Science Dept., Stanford University, Stanford, CA 94305

Abstract

Most real-world data is stored in relational form. In contrast, most statistical learning methods work with "flat" data representations, forcing us to convert our data into a form that loses much of the relational structure. The recently introduced framework of *probabilistic relational models* (PRMs) allows us to represent probabilistic models over multiple entities that utilize the relations between them. In this paper, we propose the use of probabilistic models not only for the attributes in a relational model, but for the relational structure itself. We propose two mechanisms for modeling *structural uncertainty*: *reference uncertainty* and *existence uncertainty*. We describe the appropriate conditions for using each model and present learning algorithms for each. We present experimental results showing that the learned models can be used to predict relational structure and, moreover, the observed relational structure can be used to provide better predictions for the attributes in the model.

1. Introduction

Relational models are the most common representation of structured data. Enterprise business data, medical records, and scientific datasets are all stored in relational databases. A relational model captures the set of entities in our universe, their properties, and the relationships between them. Recently, there has been growing interest in extracting information, such as patterns and regularities, from these huge amounts of data (Lavrac & Dzeroski, 1994).

Bayesian networks have been shown to provide a good representation language for statistical patterns in real-world domains. By learning a Bayesian network from data (Heckerman, 1998), we can obtain a deeper understanding of our domain and the statistical dependencies in it. A learned Bayesian network can also be used for reaching conclusions about attributes whose values may be unobserved.

Unfortunately, Bayesian networks are designed for modeling attribute-based domains, where we have a single table of IID instances. They cannot be used for modeling richer relational data sets. *Probabilistic relational models (PRMs)* are a recent development (Koller & Pfeffer, 1998; Poole, 1993) that extend the standard attribute-based Bayesian network representation to incorporate a much richer relational structure. These models allow properties of an entity to depend probabilistically on properties of other *related* entities. The model represents a generic dependence for a *class* of objects, which is then instantiated for particular sets of entities and relations between them. Friedman *et al.* (1999) adapt the machinery for learning Bayesian networks from flat data to the task of learning PRMs from structured relational data.

The PRM framework focuses on modeling the distribution over the attributes of the objects in the model. It takes the relational structure itself — the relational links between entities — to be background knowledge, determined outside the probabilistic model. This assumption implies that the model cannot be used to predict the relational structure itself. Thus, for example, we cannot use it to predict that there exists a money-laundering relation between a bank and a drug cartel. A more subtle point is that the relational structure is informative in and of itself. For example, the links from and to a web page are very informative about the type of web page (Craven et al., 1998), and the citation links between papers are very informative about the paper topics (Cohn & Hofmann, 2001).

In this paper, we provide a framework for specifying and learning a probabilistic model of the relational structure. This concept, called *structural uncertainty*, was first introduced by Koller and Pfeffer (1998). In this paper, we extend their notion of *reference uncertainty* to make it suitable for a learning framework; we also introduce a new type of structural uncertainty, called *existence uncertainty*. We present a framework for learning these models from a relational database, and present empirical results on real-world data showing that these models can be used to predict relational structure, as well as use an observed relational structure to provide better predictions about attribute values.

2. Probabilistic Relational Models

A *probabilistic relational model (PRM)* specifies a template for a probability distribution over a database. The template describes the relational schema for the domain, and the probabilistic dependencies between attributes in the domain. A PRM, together with a particular database of objects and relations, defines a probability distribution over the attributes of the objects and the relations.

Relational Schema A schema for a relational model describes a set of *classes*, $\mathcal{X} = X_1, \ldots, X_n$. Each class is associated with a set of *descriptive attributes* and a set of *reference slots*.[1] The set of descriptive attributes of a class X is denoted $\mathcal{A}(X)$. Attribute A of class X is denoted $X.A$, and its domain of values is denoted $V(X.A)$. We assume here that domains are finite, however this is not a fundamental limitation of our approach. For example, the Actor class might have the descriptive attributes *Gender*, with domain $\{$male, female$\}$.

The set of reference slots of a class X is denoted $\mathcal{R}(X)$. We use $X.\rho$ to denote the reference slot ρ of X. Each reference slot ρ is typed: the domain type of $\mathrm{Dom}[\rho] = X$ and the range type $\mathrm{Range}[\rho] = Y$, where Y is some class in $\mathcal{X}$. A slot ρ denotes a function from $\mathrm{Dom}[\rho] = X$ to $\mathrm{Range}[\rho] = Y$. For example, we might have a class Role with the reference slots *Actor* whose range is the class Actor and *Movie* whose range is the class Movie.

It is useful to distinguish between an *entity* and a *relationship*, as in entity-relationship diagrams. In our language, classes are used to represent both entities and relationships. Thus, a relationship such as Role, which relates actors to movies, is also represented as a class, with reference slots to the class Actor and the class Movie. We use $\mathcal{X}_\mathcal{E}$ to denote the set of classes that represent entities, and $\mathcal{X}_\mathcal{R}$ to denote those that represent relationships. We use the generic term *object* to refer both to entities and to relationships.

The semantics of this language is straightforward. An instantiation $\mathcal{I}$ specifies the set of objects in each class X, and the values for each attribute and each reference slots of each object. For example, Figure 1 shows an instantiation of our simple movie schema. It specifies a particular set of actors, movies and roles, along with values for each of their attributes and references.

[1]There is a direct mapping between our notion of class and the tables in a relational database: descriptive attributes correspond to standard table attributes, and reference slots correspond to foreign keys (key attributes of another table).

ACTOR

name	gender
fred	male
ginger	female
bing	male

MOVIE

name	genre
m1	drama
m2	comedy

ROLE

role	movie	actor	role-type
r1	m1	fred	hero
r2	m1	ginger	heroine
r3	m1	bing	villain
r4	m2	bing	hero
r5	m2	ginger	love-interest

Figure 1. An instantiation of the relational schema for a simple movie domain.

As discussed in the introduction, our goal in this paper is to construct probabilistic models over instantiations. To do so, we need to provide enough background knowledge to circumscribe the set of possible instantiations. Friedman *et al.* (1999) assume that the entire relational structure is given as background knowledge. In other words, they assume that they are given a *relational skeleton*, σ_r, which specifies the set of objects in all classes, as well as all the relationships that hold between them (in other words, it specifies the values for all of the reference slots). In our simple movie example, the relational skeleton would contain all of the information except for the gender of the actors, the genre of the movies, and the nature of the role.

Probabilistic Model for Attributes A probabilistic relational model Π specifies a probability distribution over all instantiations $\mathcal{I}$ of the relational schema. It consists of the qualitative dependency structure, $\mathcal{S}$, and the parameters associated with it, $\theta_\mathcal{S}$. The dependency structure is defined by associating with each attribute $X.A$ a set of *parents* $\mathrm{Pa}(X.A)$. Each parent of $X.A$ has the form $X.\tau.B$ where τ is either empty or a single slot ρ. (PRMs also allow dependencies on longer *slot chains*, but we have chosen to omit those for simplicity of presentation.) To understand the semantics of this dependence, note that $x.\tau.A$ is a multiset of values S in $V(X.\tau.A)$. We use the notion of *aggregation* from database theory to define the dependence on a multiset; thus, $x.A$ will depend probabilistically on some aggregate property $\gamma(S)$. In this paper, we use the *median* for ordinal attributes, and the *mode* (most common value) for others. When S is single-valued, both reduces to a dependence on the value of $x.\tau.B$.

The quantitative part of the PRM specifies the parameterization of the model. Given a set of parents for an attribute, we can define a local probability model by associating with it a *conditional probability distribution (CPD)*. For each attribute we have a CPD that specifies $P(X.A \mid \mathrm{Pa}(X.A))$.

Definition 1: A *probabilistic relational model (PRM)* Π for a relational schema $\mathcal{S}$ is defined as follows. For each class $X \in \mathcal{X}$ and each descriptive attribute $A \in \mathcal{A}(X)$, we have a set of *parents* $\mathrm{Pa}(X.A)$, and a *conditional probability distribution (CPD)* that represents $P_\Pi(X.A \mid \mathrm{Pa}(X.A))$. ∎

Given a relational skeleton σ_r, a PRM Π specifies a distribution over a set of instantiations $\mathcal{I}$ consistent with σ_r:

$$P(\mathcal{I} \mid \sigma_r, \Pi) = \prod_{x \in \sigma_r(X)} \prod_{A \in \mathcal{A}(x)} P(x.A \mid \mathrm{Pa}(x.A)) \quad (1)$$

where $\sigma_r(X)$ are the objects of each class as specified by the relational skeleton σ_r (in general we will use the notation $\sigma(X)$ to refer to the set objects of each class as defined by any type of domain skeleton).

For this definition to specify a coherent probability distribution over instantiations, we must ensure that our probabilistic dependencies are acyclic, so that a random variable does not depend, directly or indirectly, on its own value. Moreover, we want to guarantee that this will be the case for *any* skeleton. For this purpose, we use a *class dependency graph*, which describes all possible dependencies among attributes. In this graph, we have an (intra-object) edge $X.B \rightarrow X.A$ if $X.B$ is a parent of $X.A$. If $X.\rho.B$ is a parent of $X.A$, and $Y = \mathrm{Range}[\rho]$, we have an (inter-object) edge $Y.B \rightarrow X.A$. If the dependency graph of $\mathcal{S}$ is acyclic, then it defines a legal model for any relational skeleton σ_r (Friedman et al., 1999).

3. Structural Uncertainty

In the model described in the previous section, all relations between attributes are determined by the relational skeleton σ_r; only the descriptive attributes are uncertain. Thus, Eq. (1) determines the probabilistic model of the attributes of objects, but does not provide a model for the relations between objects. In this section, we extend our probabilistic model to allow for *structural uncertainty*. Here, we do not treat the relational structure as background knowledge, but choose to model it explicitly within the probabilistic framework. Clearly, there are many ways to represent a probability distribution over the relational structure. In this paper, we explore two simple yet natural models: *Reference Uncertainty* and *Existence Uncertainty*.

Reference Uncertainty In this model, we assume that the objects are prespecified, but relations among them, i.e., reference slots, are subject to random choices. Thus, rather than being given a full relational skeleton σ_r, we assume that we are given an *object skeleton* σ_o. The object skeleton specifies only the objects $\sigma_o(X)$ in each class $X \in \mathcal{X}$, but not the values of the reference slots. In our example above, the object skeleton would specify only the set of movies, actors, and roles in the database: $\sigma_o(\mathsf{Actor}) = \{\text{fred, ginger, bing}\}$, $\sigma_o(\mathsf{Movie}) = \{m1, m2\}$, and $\sigma_o(\mathsf{Role}) = \{r1, r2, r3, r4, r5\}$. In this case, we must specify a probabilistic model for the value of the reference slots $X.\rho$. The domain of a reference slot $X.\rho$ is the set of keys (unique identifiers) of the objects in the class Y to which $X.\rho$ refers. Thus, we need to specify a probability distribution over the set of all objects in Y.

A naive approach is to simply have the PRM specify

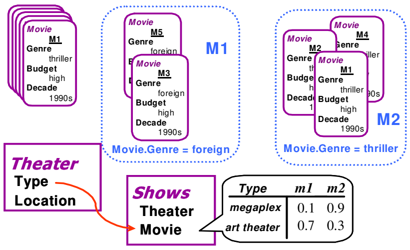

Figure 2. A simple example of reference uncertainty

a probability distribution directly over $\sigma_o(Y)$. This approach has two major flaws. Most obviously, this distribution would require a parameter for each object in Y. More importantly, we want our dependency model to be general enough to apply over all possible object skeletons σ_o; a distribution defined in terms of the objects within a specific object skeleton would not apply to others.

We achieve a general and compact representation by partitioning the class Y into subsets according to the values of some of its attributes. We assume that the value of $X.\rho$ is chosen by first selecting a partition, and then selecting an object within that partition uniformly. For example, as shown in Figure 2, we can partition the class Movie by *Genre*, indicating that a movie theater first selects which genre of movie it wants to show, and then uniformly among the movies with the selected genre. The decision on genre might depend on the type of theater.

We make this intuition precise by defining, for each slot ρ, a set of *partition attributes* $Partition[\rho] \subseteq \mathcal{A}(Y)$. In the above example, $Partition[\mathit{Shows}] = \{\mathit{Genre}\}$. We now need to specify the distribution that the reference value of ρ falls into one partition versus another. We accomplish this by introducing S_ρ as a new attribute of X, called a *selector attribute*; it takes on a value v in the space of possible instantiations $V(Partition[\rho])$. Each possible value v determines a subset of Y from which the value of ρ (the referent) will be selected. We use Y_v to represent the resulting partition of $\sigma_o(Y)$.

We now represent a probabilistic model over the values of ρ by specifying how likely it is to reference objects in one subset in the partition versus another. We define a probabilistic model for the selector attribute S_ρ. This model is the same as that of any other attribute: it has a set of parents and a CPD. Thus, the CPD for S_ρ would specify a probability distribution over possible instantiations. As for descriptive attributes, we allow the distribution of the slot to depend on other aspects of the domain. For example, an independent movie theater may be more likely to show foreign movies while a megaplex may prefer to show thrillers. Thus, the CPD of $\mathsf{Show}.S_{\mathit{Movie}}$ might have as a parent $\mathsf{Theater}.\mathit{Type}$. The choice of value for S_ρ determines the partition Y_v from which the reference value of ρ is chosen; the choice of reference value for ρ is uniformly distributed within this set.

Definition 2: A *probabilistic relational model* Π *with reference uncertainty* has the same components as in Definition 1. In addition, for each reference slot $\rho \in \mathcal{R}(X)$ with $\mathrm{Range}[\rho] = Y$, we have:

- a set of attributes $Partition[\rho] \subseteq \mathcal{A}(Y)$;
- a new selector attribute S_ρ within X which takes on values in the cross-product space $V(Partition[\rho])$;
- a set of parents and a CPD for S_ρ. ∎

To define the semantics of this extension, we must define the probability of reference slots as well as descriptive attributes:

$$P(\mathcal{I} \mid \sigma_o, \Pi) = \prod_{x \in \sigma_o(X)} \prod_{A \in \mathcal{A}(x)} P(x.A \mid \mathrm{Pa}(x.A)) \prod_{\rho \in \mathcal{R}(x), \mathrm{Range}[\rho]=Y} \frac{P(x.S_\rho = v[x.\rho] \mid \mathrm{Pa}(x.S_\rho))}{|Y_v|} \quad (2)$$

where we take $v[x.\rho]$ to refer to the instantiation v of the attributes $Partition[\rho]$ for the object $x.\rho$ in the instantiation $\mathcal{I}$. Note that the last term in Eq. (2) depends on $\mathcal{I}$ in three ways: the interpretation of $x.\rho$, the values of the attributes $\Psi[\rho]$ within the object $x.\rho$, and the size of Y_v.

As above, we must guarantee that dependencies are acyclic for every object skeleton. We accomplish this goal by extending our definition of class dependency graph. The graph has a node for each descriptive or selector attribute $X.A$ and the following edges:

- For any descriptive or selector attribute $X.C$ and any of its parents $X.\tau.B$ we introduce an edge from $Y.B$ to $X.C$, where $Y = \mathrm{Range}[\tau]$.
- For any descriptive or selector attribute C, and any of its parents $X.\rho.B$ we add an edge from $X.S_\rho$ to $X.C$.
- For each slot $X.\rho$, and each $Y.B \in Partition[\rho]$ (for $Y = \mathrm{Range}[\rho]$), we add an edge $Y.B \rightarrow X.S_\rho$.

The first class of edges in this definition is identical to the definition of dependency graph above, except that it deals with selector as well as descriptive attributes. Edges of the second type reflect the fact that the specific choice of parent for a node depends on the reference value of the parent slot. Edges of the third type represent the dependency of a slot on the attributes of the associated partition. To see why this is required, we observe that our choice of reference value for $x.\rho$ depends on the values of the partition attributes $Partition[X.\rho]$ of all of the different objects in Y. Thus, these attributes must be determined before $x.\rho$ is determined. In our example, as $Partition[Shows] = \{Genre\}$, the genres of all movies must be determined before we can select the value of the reference slot *Shows*.

Once again, we can show that if this dependency graph is acyclic, it defines a coherent probabilistic model.

Theorem 3: *Let* Π *be a PRM with relational uncertainty and acyclic dependency graph. Let* σ_o *be an object skeleton. Then* Π *and* σ_o *uniquely define a probability distribution over instantiations* $\mathcal{I}$ *that extend* σ_o *via Eq. (2).*

Existence Uncertainty The reference uncertainty model of the previous section assumes that the number of objects is known. Thus, if we consider a division of objects into entities and relations, the number of objects in classes of both types are fixed. In this section, we consider models where the number of relationship objects is not fixed in advance. Consider a simple citation domain with an entity class Paper and a relation class Cite. In this case, we might assume that the set of papers is part of our background knowledge, but we want to provide an explicit model for the presence or absence of citations. More generally, we assume that we are given only an *entity skeleton* σ_e, which specifies the set of objects in our domain only for the entity classes. In our example of Figure 1, the entity skeleton would include only the set of actors and movies. We call the entity classes *determined* and the others *undetermined*. We note that relationship classes typically represent many-many relationships; they have at least two reference slots, which refer to determined classes. For example, our Cite class would have reference slots *Citing-Paper* and *Cited-Paper*.

Our basic approach is to allow objects whose existence is uncertain — the objects in the undetermined classes. One way of achieving this effect is by introducing into the model all of the entities that can *potentially* exist in it; with each of them we associate a special binary variable that tells us whether the entity actually exists or not. Note that this construction is purely conceptual; we never explicitly construct a model containing non-existent objects. In our example above, the domain of the Cite class in a given instantiation $\mathcal{I}$ is $\mathcal{I}(\mathsf{Paper}) \times \mathcal{I}(\mathsf{Paper})$. Each "potential" object $x = \mathsf{Cite}(y_1, y_2)$ in this domain is associated with a binary attribute $x.E$ that specifies whether paper y_1 did or did not cite in paper y_2.

Definition 4: We define an *undetermined* class X as follows. Let $\rho_1, \ldots, \rho_k$ be the set of reference slots of X, and let $Y_i = \mathrm{Range}[\rho_i]$. In any instantiation $\mathcal{I}$, we require that $\mathcal{I}(X) = \mathcal{I}(Y_1) \times \cdots \mathcal{I}(Y_k)$. For $(y_1, \ldots, y_k) \in \mathcal{I}(Y_1) \times \cdots \mathcal{I}(Y_k)$, we use $X[y_1, \ldots, y_k]$ to denote the corresponding object in X. Each X has a special *existence* attribute $X.E$ whose values are $V(E) = \{\mathit{true}, \mathit{false}\}$. For uniformity of notation, we introduce an E attribute for all classes; for classes that are determined, the E value is defined to be always *true*. We require that all of the reference slots of a determined class X have a range type which is also a determined class. ∎

The existence attribute for an undetermined class is treated in the same way as a descriptive attribute in our dependency model, in that it can have parents and children, and is associated with a CPD. In our citation domain, it is reasonable to assume that the existence of a citation may depend on the topic of the citing paper and the topic of the cited paper (it is more likely that citations will exist between papers with the same topic). Our definitions are such that the semantics of the model does not change. By defining the existence events to be attributes, and incorporating them appropriately into the probabilistic model, we have set things up so that Eq. (1) applies unchanged.

We must, however, place some restrictions on our model to ensure that our definitions lead to a coherent probability model. For example, if the range type of a slot of an undetermined class refers to itself ($\mathrm{Range}[X.\rho] = X$) then the set $\mathcal{I}(X)$ is defined circularly, in terms of itself. We say that an undetermined class X is *coherent* if it satisfies the following restrictions: (1) An attribute $X.A$ cannot be an ancestor of $X.E$. (2) An object can only exist if all the objects it refers to exist, i.e., for every slot $\rho \in \mathcal{R}(X)$, $P(x.E = \mathit{false} \mid x.\rho.E = \mathit{false}) = 1$. (3) Dependencies can only "pass through" objects that exist. More precisely, for any slot $Y.\rho$ of range-type X, we define the *usable slot* $\overline{\rho}$ as follows: for any $y \in \mathcal{I}(Y)$, we define $y.\overline{\rho} = \{x \in y.\rho \,:\, x.E = \mathit{true}\}$. We allow only $\overline{\rho}$ to be used as a parent in the dependency model $\mathcal{S}$.

We can use our class dependency graph to capture most of these requirements. For every $X.A$, we introduce an edge from $X.E$ to $X.A$. For every slot $\rho \in \mathcal{R}(X)$ whose range type is Y, we have an edge from $Y.E$ to $X.E$. For every attribute $X.A$ and every $X.\overline{\rho}.B \in \mathrm{Pa}(X.A)$, we have an edge from $\mathrm{Range}[\rho].E$ to $X.A$. As before, we require that the attribute dependency graph is acyclic. It turns out that our requirements are sufficient to guarantee that every undetermined class is coherent, and to allow our extended language to be viewed as a standard PRM.

Theorem 5: *Let Π be a PRM with undetermined classes and an acyclic class dependency graph. Let σ_e be an entity skeleton. Then the PRM and σ_e uniquely define a relational skeleton σ_r over all classes, and a probability distribution over instantiations $\mathcal{I}$ that extends σ_e via Eq. (1).*

Note that a full instantiation $\mathcal{I}$ also determines the existence attributes for undetermined classes. Hence, the probability distribution induced by the PRM also specifies the probability that a certain entity will exist in the model.

We note that real-world databases do not specify the descriptive attributes of entities that do not exist. However, since we only allow dependencies on objects that exist (for which $x.E = \mathit{true}$), then nonexistent objects are *leaves* in the model and can be ignored in the computation of $P(\mathcal{I} \mid \sigma_e, \Pi)$. The only contribution of a nonexistent entity x to the probability of an instantiation $\mathcal{I}$ is the probability that $x.E = \mathit{false}$.

Example: Word models Our two models of structural uncertainty induce simple yet intuitive models for link existence. We illustrate this by showing a natural connection to the two most common models of word appearance in documents. Suppose our domain contains two entity classes: Document, representing the set of documents in our corpus, and Words, representing the words contained in our dictionary. Documents may have descriptive attributes such as *Topic*; dictionary entries would have the attribute *Word*, which is the word itself, and may also have additional attributes such as the type of word. The relationship class Appearance represents the appearance of words in documents; it has two slots *InDoc* and *HasWord*. In this schema, structural uncertainty corresponds to a probabilistic model of the appearance of words in documents.

In existence uncertainty, the class Appearance is an an undetermined class; the potential objects in this class correspond to document-word pairs (d, w), and the assertion $\mathsf{Appearance}(d, w).E = \mathit{true}$ means that the particular dictionary entry w appears in the particular document d. Now, suppose that $\mathsf{Appearance}.E$ has the parents Appearance.*InDoc*.*Topic* and Appearance.*HasWord*.*Word*. This implies, that, for each word w and topic t, we have a parameter $p_{w,t}$ which is the probability that a word w appears in a document of topic t. Furthermore, the different events $\mathsf{Appearance}(d, w).E$ are conditionally independent given the topic t. It is easy to see that this model is equivalent to the model often called *binary naive Bayes model* (McCallum & Nigam, 1998), where the class variable is the topic and the conditionally independent features are binary variables corresponding to the appearance of different dictionary entries in the document.

When using reference uncertainty, we can consider several modeling alternatives. The most straightforward model is to view a document as a bag of words. Now, Appearance also includes an attribute that designates the position of the word in the document. Thus, a document of n words has n related Appearance objects. We can provide a probabilistic model of word appearance by using reference uncertainty over the slot Appearance.*HasWord*. In particular, if we choose $Partition[\mathsf{Appearance}.\mathit{HasWord}] = \mathsf{Word}.\mathit{Word}$, then we have a multinomial distribution over the words in the dictionary. If we set Appearance.*InDoc*.*Topic* as the parent of the selector variable $\mathsf{Appearance}.S_{\mathit{HasWord}}$, then we get a different multinomial distribution over words for each topic. The result is a model where a document is viewed as a sequence of independent samples from a multinomial distribution over the dictionary, where the sample distribution depends on the document topic. This document model is called the *multinomial Naive Bayesian model* (McCallum & Nigam, 1998).

Thus, for this simple PRM structure, the two forms of structural uncertainty lead to models that are well-studied within the statistical NLP community. However, the language of PRMs allows us to represent more complex structures: Both the existence and reference uncertainty can depend on properties of words rather than on the exact identity of the word; they can also depend on other attributes, such as the research area of the document's author.

4. Learning PRMs

In the previous sections we discussed three variants of PRM models that differ in their expressive power. Our aim is to *learn* such models from data: given a schema and an instance, construct a PRM that describes the dependencies between objects in the schema. We stress that, all three PRM model variants are learned using the same type of training data: a complete instantiation that describes a set of objects, their attribute values and their reference slots. However, in each variant, we attempt to learn somewhat different structure from this data. For basic PRMs, we learn the probability of attributes given other attributes; for PRMs with reference uncertainty, we also attempt to learn the rules that govern the choice of slot references; and for PRMs with existence uncertainty, we attempt to learn the probability of existence of relationship objects.

We separate the learning problem into two questions: evaluating the "goodness" of a candidate structure, and searching the space of legal candidate structures.

Model Scoring For scoring candidate structures, we adapt Bayesian *model selection* (Heckerman, 1998). We compute the posterior probability of a structure $\mathcal{S}$ given an instantiation $\mathcal{I}$. Using Bayes rule we have that $P(\mathcal{S} \mid \mathcal{I}, \sigma) \propto P(\mathcal{I} \mid \mathcal{S}, \sigma) P(\mathcal{S} \mid \sigma)$. This score is composed of two main parts: the prior probability of $\mathcal{S}$, and the probability of the instantiation assuming the structure is $\mathcal{S}$. By making fairly reasonable assumptions about the prior probability of structures and parameters, this term can be *decomposed* into a product of terms. Each term in the decomposed form measures how well we predict the values of $X.A$ given the values of its parents. Moreover, the term for $P(X.A \mid \mathbf{u})$ depends only on the *sufficient statistics* $C_{X.A}[v, \mathbf{u}]$, that count the number of entities with $x.A = v$ and Pa$(x.A) = \mathbf{u}$.

The extension of the Bayesian score to PRMs with existence uncertainty is straightforward. The only new issue is how to compute sufficient statistics that include existence attributes $x.E$ without explicitly enumerating all the non-existent entity. We perform this computation by counting, for each possible instantiation of Pa$(X.E)$, the number of potential objects with that instantiation, and subtracting the actual number of objects x with that parent instantiation.

The extension required to deal with reference uncertainty is also not a difficult one. Once we fix the set partition attributes $\Psi[\rho]$, a CPD for S_ρ compactly defines a distribution over values of ρ. Thus, scoring the success in predicting the value of ρ can be done efficiently using standard Bayesian methods used for attribute uncertainty (e.g. using a standard Dirichlet prior over values of ρ).

Model Search To find a high-scoring structure, we use a simple search procedure that considers operators such as adding, deleting, or reversing edges in the dependency model $\mathcal{S}$. The procedure performs greedy hill-climbing search, using the Bayesian score to evaluate structures.

No extensions to the search algorithm are required to handle existence uncertainty. We simply introduce the new attributes $X.E$, and integrate them into the search space, as usual. As usual, we enforce coherence using the class dependency graph described above.

The extension for incorporating reference uncertainty is more subtle. Initially, the partition of the range class for a slot $X.\rho$ is not given in the model. Therefore, we must also search for the appropriate set of attributes $\Psi[\rho]$. We introduce two new operators **refine** and **abstract**, which modify the partition by adding and deleting attributes from $\Psi[\rho]$. Initially, $\Psi[\rho]$ is empty for each ρ. The **refine** operator adds an attribute into $\Psi[\rho]$; the **abstract** operator deletes one. These newly introduced operators are treated by the search algorithm in exactly the same way as the standard edge-manipulation operators: the change in the score is evaluated for each possible operator, and the algorithm selects the best one to execute.

We note that, as usual, the decomposition of the score can be exploited to substantially speed up the search. In general, the score change resulting from an operator ω is re-evaluated only after applying an operator ω' that modifies the parent or partition set of an attribute that ω modifies. This is also true when we consider operators that modify the parent of selector attributes and existence attributes.

5. Results

We evaluated the methods on several real-life data sets, comparing standard PRMs, PRMs with reference uncertainty (RU), and PRMs with existence uncertainty (EU). Our experiments used the Bayesian score with a uniform Dirichlet parameter prior with equivalent sample size $\alpha = 2$, and a uniform distribution over structures.

We first tested whether the additional expressive power allows us to better capture regularities in the domain. Toward this end, we evaluated the likelihood of test data given our learned models. Unfortunately, we cannot directly compare likelihoods, since the PRMs involve different sets of probabilistic events. Instead, we compare the two variants of PRMs with structural uncertainty, EU and RU, to "baseline" models which incorporate link probabilities, but make the "null" assumption that the link structure is uncorrelated with the descriptive attributes. For reference uncertainty, the baseline has $\Psi[\rho] = \emptyset$ for each slot. For existence uncertainty, it forces $x.E$ to have no parents in the model.

We evaluated these different variants on a dataset that combines information about movies and actors from the

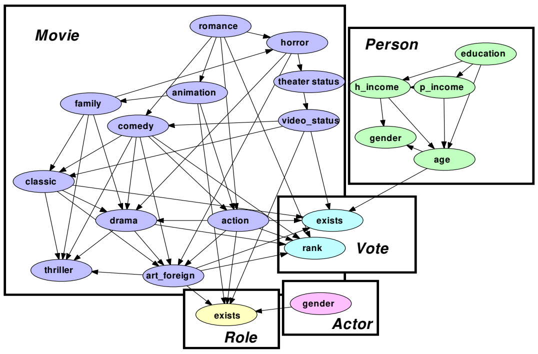

Figure 3. The PRM learned using existence uncertainty.

Internet Movie Database[2] and information about people's ratings of movies from the Each Movie dataset,[3] where each person's demographic information was extended with census information for their zipcode. From these, we constructed five classes (with approximate sizes shown): Movie (1600), Actor (35,000); Role (50,000), Person (25,000), and Vote (300,000).

We modeled uncertainty about the link structure of the classes Role (relating actors to movies) and Vote (relating people to movies). This was done either by modeling the probability of the existence of such objects, or modeling the reference uncertainty of the slots of these objects. We trained on nine-tenths of the data and evaluated the log-likelihood of the held-out test set. Both models of structural uncertainty significantly outperform their "baseline" counterparts. In particular, we obtained a log-likelihood of $-210,044$ for the EU model, as compared to $-213,798$ for the baseline EU model. For RU, we obtained a log-likelihood of $-149,705$ as compared to $-152,280$ for the baseline model. Thus, we see that the model where the relational structure is correlated with the attribute values is substantially more predictive than the baseline model that takes them to be independent: although any particular link is still a low-probability event, our structural uncertainty models are much more predictive of its presence.

Figure 3 shows the EU model learned. We learned that the existence of a vote depends on the age of the voter and the movie genre, and the existence of a role depends on the gender of the actor and the movie genre. In the RU model (figure omitted due to space constraints), we partition each of the movie reference slots on genre attributes; we partition the actor reference slot on the actor's gender; and we partition the person reference of votes on age, gender and education. An examination of the models shows, for example, that younger voters are much more likely to have voted on action movies and that male action movies roles are more likely to exist than female roles.

Next, we considered the conjecture that by modeling link

[2] © 1990-2000 Internet Movie Database Limited.

[3] http://www.research.digital.com/SRC/EachMovie.

Table 1. Prediction accuracy of topic/category attribute of documents in the Core and WebKB datasets. Accuracies and reported standard deviations are based on a 10-fold cross validation.

	Cora	WebKB
baseline	75 ± 2.0	74 ± 2.5
RU Citing	81 ± 1.7	78 ± 2.3
RU Cited	79 ± 1.3	77 ± 1.5
EU	85 ± 0.9	82 ± 1.3

structure we can improve the prediction of descriptive attributes. Here, we hide some attribute of a test-set object, and compute the probability over its possible values given the values of other attributes on the one hand, or the values of other attributes and the link structure on the other. We tested on two similar domains: Cora (McCallum et al., 2000) and WebKB (Craven et al., 1998). The Cora dataset contains 4000 machine learning papers, each with a seven-valued *Topic* attribute, and 6000 citations. The WebKB dataset contains approximately 4000 pages from several Computer Science departments, with a five-valued attribute representing their "type", and 10,000 links between web pages. In both datasets we also have access to the content of the document (webpage/paper), which we summarize using a set of attributes that represent the presence of different words on the page (a binary Naive Bayes model). After stemming and removing stop words and rare words, the dictionary contains 1400 words in the Cora domain, and 800 words in the WebKB domain.

In both domains, we compared the performance of models that use only word appearance information to predict the category of the document with models that also used probabilistic information about the link from one document to another. We fixed the dependency structure of the models, using basically the same structure for both domains. In the Cora EU model, the existence of a citation depends on the topic of the citing paper and the cited paper. We evaluated two symmetrical RU models. In the first, we partition the citing paper by topic, inducing a distribution over the topic of Citation.*Citing*. The parent of the selector variable is Citation.*Cited*.*Topic*. The second model is symmetrical, using reference uncertainty over the cited paper.

Table 1 shows prediction accuracy on both data sets. We see that both models of structural uncertainty significantly improve the accuracy scores, although existence uncertainty seems to be superior. Interestingly, the variant of the RU model that models reference uncertainty over the citing paper based on the topics of papers cited (or the from webpage based on the categories of pages to which it points) outperforms the cited variant. However, in all cases, the addition of citation/hyperlink information helps resolve ambiguous cases that are misclassified by the baseline model that considers words alone. For example, paper #506 is a Probabilistic Methods paper, but is classified based on its words as a Genetic Algorithms paper (with probability 0.54). However, the paper cites two Probabilistic Methods papers, and is cited by three Probabilistic Methods papers, leading both the EU and RU models to classify it correctly. Paper #1272 contains words such as rule, theori, refin, induct, decis, and tree. The baseline model classifies it as a Rule Learning paper (probability 0.96). However, this paper cites one Neural Networks and one Reinforcement Learning paper, and is cited by seven Neural Networks, five Case-Based Reasoning, fourteen Rule Learning, three Genetic Algorithms, and seventeen Theory papers. The Cora EU model assigns it probability 0.99 of being a Theory paper, which is the correct topic. The first RU model assigns it a probability 0.56 of being Rule Learning paper, whereas the symmetric RU model classifies it correctly. We explain this phenomenon by the fact that most of the information in this case is in the topics of citing papers; it appears that RU models can make better use of information in the parents of the selector variable than in the partitioning variables.

6. Discussion and Conclusions

In this paper, we present two representations for structural uncertainty: reference uncertainty and existence uncertainty. Reference uncertainty models the process by which reference slots are selected from a given set. Existence uncertainty provides a model for whether a relation exists between two objects. We have shown how to integrate them with our learning framework, and presented results showing that they allow interesting patterns to be learned. The ability to learn probabilistic models of relational structure has many applications. It allows us to predict whether two objects with given properties are more likely to be related to each other. More surprisingly, the link structure also allows us to predict attribute values of interest. For example, we can better predict the topic of a paper by using the fact that it cites certain types of papers.

Several recent works in the literature examine learning from relational data. Kleinberg (1998) learns a global property of a relation graph ("authority" of web pages) based on local connectivity. This approach does not generalize beyond the training data, and ignores attributes of the pages (e.g., words). Slattery and Mitchell (2000) integrate Kleinberg's authority recognition module with a first-order rule learner to perform classification that also utilizes the relational structure in the test set. Their approach is intended purely for classification, and is not a statistical model of the domain. Furthermore, their approach is not based on a single coherent framework, so that the results of two different modules are combined procedurally.

A stochastic relational model recently defined by Cohn and Hofmann (2001) introduces a *latent (hidden)* variable that describes the "class" of each document. Their model assumes that word occurrences and links to other documents are independent given the document's class. This model is similar to a PRM model with reference uncertainty, but differs from it in several important ways. First, it uses a multinomial distribution over specific citations, preventing the model from generalizing to a different test set. Second, each paper is assumed to be independent, so there is no ability to reach conclusions about the topic of a cited paper from that of a citing paper. Finally, dependencies between the words appearing in the document and the presence or absence of a citation cannot be represented.

The ability to learn probabilistic models of relational structure is an exciting new direction for machine learning. Our treatment here only scratches the surface of this area. In particular, although useful, neither of the representations proposed for structural uncertainty is entirely satisfying as a generative model. Furthermore, both models are restricted to considering the probabilistic model of a single relational "link" in isolation. These simple models can be seen as the naive Bayes of structural uncertainty; in practice, relational patterns involve multiple links, e.g., the concepts of hubs and authorities. In future work, we hope to provide a unified framework for representing and learning probabilistic models of relational "fingerprints" involving multiple entities and links.

Acknowledgments This work was supported by ONR contract N66001-97-C-8554 under DARPA's HPKB program. N. Friedman was also supported by Israel Science Foundation grant 244/99 and an Alon Fellowship.

References

Cohn, D., & Hofmann, T. (2001). The missing link—a probabilistic model of document content and hypertext connectivity. *Proc. NIPS 13*. To appear.

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slattery, S. (1998). Learning to extract symbolic knowledge from the world wide web. *Proc. AAAI.*

Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999). Learning probabilistic relational models. *Proc. IJCAI.*

Heckerman, D. (1998). A tutorial on learning with Bayesian networks. In M. I. Jordan (Ed.), *Learning in graphical models*. Cambridge, MA: MIT Press.

Kleinberg, J. (1998). Authoritative sources in a hyperlinked environment. *Proc. SODA*.

Koller, D., & Pfeffer, A. (1998). Probabilistic frame-based systems. *Proc. AAAI.*

Lavrac, N., & Dzeroski, S. (1994). Inductive logic programming: Techniques and applications.

McCallum, A., & Nigam, K. (1998). A comparison of event models for naive Bayes text classification. *AAAI-98 Workshop on Learning for Text Categorization*.

McCallum, A., Nigam, K., Rennie, J., & Seymore, K. (2000). Automating the construction of internet portals with machine learning. *Information Retrieval*, *3*, 127–163.

Poole, D. (1993). Probabilistic Horn abduction and Bayesian networks. *Artificial Intelligence*, *64*, 81–129.

Slattery, S., & Mitchell, T. (2000). Discovering test set regularities in relational domains. *Proc. ICML.*