Learning Probabilistic Models of Link Structure

Lise Getoor GETOOR@CS.UMD.EDU
Computer Science Dept.
University of Maryland
College Park, MD 20742

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

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

Ben Taskar BTASKAR@CS.STANFORD.EDU
Computer Science Dept.
Stanford University
Stanford, CA 94305

Abstract

Most real-world data is heterogeneous and richly interconnected. Examples include the Web, hypertext, bibliometric data and social networks. 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 link structure. The recently introduced framework of *probabilistic relational models* (PRMs) embraces the object-relational nature of structured data by capturing probabilistic interactions between attributes of related entities. In this paper, we extend this framework by modeling interactions between the attributes and the link structure itself. An advantage of our approach is a unified generative model for both content and relational structure. We propose two mechanisms for representing a probabilistic distribution over link structures: *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 link structure and, moreover, the observed link structure can be used to provide better predictions for the attributes in the model.

1. Introduction

In recent years, we have witnessed an explosion in the amount of information that is available to us in digital form. More and more data is being stored, and more and more data is being made accessible, through traditional interfaces such as corporate databases and, of course, via the Internet and the World Wide Web. There is much to be gained by applying machine learning techniques to these data, in order to extract useful information and patterns.

Most often, the objects in these data do not exist in isolation — there are "links" or relationships that hold between them. For example, there are links from one web page to another, a scientific

paper cites another paper, and an actor is linked to a movie by the appearance relationship. Most work in machine learning, however, has focused on "flat" data, where the instances are independent and identically distributed. The main exception to this rule has been the work on inductive logic programming (Muggleton, 1992, Lavrač and Džeroski, 1994). This work focuses on the problem of inferring link structure from a logical perspective.

More recently, there has been a growing interest in combining the statistical approaches that have been so successful when learning from a collection of homogeneous independent instances (typically represented as a single table in a relational database) with relational learning methods. Slattery and Craven (1998) were the first to consider combining a statical approach with a first-order relational learner (FOIL) for the task of web page classification. Chakrabarti et al. (1998) explored methods for hypertext classifications which used both the content of the current page and information from related pages. Popescul et al. (2002) use an approach that uses a relational learner to guide in the construction of features to be used by a (statistical) propositional learner. Yang et al. (2002) identify certain categories of relational regularities and explore the conditions under which they can be exploited to improve classification accuracy.

Here, we propose a unified statistical framework for content and links. Our frameworks builds on the recent work on *probabilistic relational models (PRMs)* (Poole, 1993, Ngo and Haddawy, 1995, Koller and Pfeffer, 1998). PRMs 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 a set of unrelated homogeneous instances to the task of learning PRMs from structured relational data.

The original PRM framework focused on modeling the distribution over the attributes of the objects in the model. It took 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. A more subtle yet very important 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 and Hofmann, 2001).

The PRM framework can be naturally extended to address this limitation. By making links first-class citizens in the model, the PRM language easily allows us to place a probabilistic model directly over them. In other words, we can extend our framework to define probability distributions over the presence of relational links between objects in our model. The concept of a probabilistic model over relational structure was introduced by Koller and Pfeffer (1998) under the name *structural uncertainty*. They defined several variants of structural uncertainty, and presented algorithms for doing probabilistic inference in models involving structural uncertainty.

In this paper, we show how a probabilistic model of relational structure can be learned directly from data. Specifically, we provide two simple probabilistic models of link structure: The first is an extension of the *reference uncertainty* model of Koller and Pfeffer (1998), which makes it suitable for a learning framework; the second is a new type of structural uncertainty, called *existence uncertainty*. We present a clear semantics for these extensions, and propose a method for learning

such models from a relational database. We present empirical results on real-world data, showing that these models can be used to predict the link structure, as well as use the presence of (observed) links in the model to provide better predictions about attribute values. Interestingly, these benefits are obtained even with the very simple models of link structure that we use in this paper. Thus, even simplistic models of link uncertainty provide us with increased predictive accuracy.

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.

2.1 Relational Schema

A schema $\mathcal{S}$ 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 $V(X.A)$. For example, the Actor class might have the descriptive attributes *Gender*, with domain $\{$male, female$\}$. For simplicity, we assume in this paper that attribute domains are finite; this is not a fundamental limitation of our approach.

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. We note that the functional nature of slots does not prevent us from having many-to-many relations between classes. We simply use a standard transformation where we introduce a class corresponding to the relationship object. This class will have an object for every pair (or tuple) of related objects, with functional reference slots to the objects it relates. The Role class above is an example of this transformation.

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 complete instantiation $\mathcal{I}$ specifies the set of objects in each class X, and the values for each attribute and each reference slot of each object. Thus, a complete instantiation $\mathcal{I}$ is a set of objects with no missing values and no dangling references. It describes the set of objects, the relationships that hold between the objects and all the values of the attributes of the objects. 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. More precisely, 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.

2.2 Probabilistic Model for Attributes

A probabilistic relational model Π specifies a probability distribution over a set of instantiations $\mathcal{I}$ of the relational schema. More precisely, given a relational skeleton σ_r, it specifies a distribution over all complete instantiations $\mathcal{I}$ that extend the skeleton σ_r.

A PRM consists of a qualitative dependency structure, $\mathcal{G}$, and the parameters associated with it, $\theta_{\mathcal{G}}$. The dependency structure is defined by associating with each attribute $X.A$ a set of *formal parents* $\mathrm{Pa}(X.A)$. These correspond to *formal* parents; they will be instantiated in different ways for different objects in X. Intuitively, the parents are attributes that are "direct influences" on $X.A$. The attribute $X.A$ can depend on another probabilistic attribute B of X. It can also depend on attributes of related objects $X.\rho.B$.[2]

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 $P(X.A \mid \mathrm{Pa}(X.A))$. Each CPD in our PRM is *legal*, i.e., the entries are positive and sum to 1.

Definition 1 *A* probabilistic relational model (PRM) Π *for a relational schema $\mathcal{S}$ defines for each class $X \in \mathcal{X}$ and each descriptive attribute $A \in \mathcal{A}(X)$, a set of* formal parents *$Pa(X.A)$, and a* conditional probability distribution (CPD) *that represents $P(X.A \mid Pa(X.A))$.* ∎

2. PRMs allow a much richer dependency model, where objects can depend on each other via longer *slot chains*. They also allow dependencies via *inverse slots*, which generally define one-to-many relations (e.g., the set of all roles of an actor). There is no difficulty adding these features to our language, but they complicate the presentation considerably; we have therefore chosen to omit them for simplicity.

Given a relational skeleton σ_r, a PRM Π specifies a distribution over a set of instantiations $\mathcal{I}$ consistent with σ_r. This specification is done by mapping the dependencies in the class-level PRM to the actual objects in the domain. For a class X, we use $\sigma_r(X)$ to denote the objects X, 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.) Let $X.A$ be an attribute in the schema, and let x be some object in $\sigma_r(X)$. Recall that the PRM allows two types of formal parents: $X.B$ and $X.\rho.C$. For a formal parent of the form $X.B$, the corresponding actual parent of $x.A$ is $x.B$. For a formal parent of the form $X.\rho.C$, the corresponding formal parent of $x.A$ is $y.C$, where $y = x.\rho$ in σ_r. Thus, the class-level dependencies in the PRM are instantiated according to the relational skeleton, to define object-level dependencies. The parameters specified by the PRM are used for each object in the skeleton, in the obvious way.

Thus, for a given skeleton, the PRM basically defines a *ground Bayesian network*. The qualitative structure of the network is defined via an *instance dependency graph* G_{σ_r}, whose nodes correspond to descriptive attributes $x.A$ of entities in the skeleton. These are the random variables in our model. We have a directed edge from $y.B$ to $x.A$ if $y.B$ is an actual parent of $x.A$, as defined above. The quantitative parameters of the network are defined by the CPDs in the PRM, with the same CPD used multiple times in the network. This ground Bayesian network leads to the following *chain rule* which defines a distribution over the instantiations compatible with our particular skeleton σ_r:

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

For this definition to specify a coherent probability distribution over instantiations, we must ensure that our probabilistic dependencies are *acyclic*. In particular, we must verify that each random variable $x.A$ does not depend, directly or indirectly, on its own value. In other words, G_{σ_r} must be acyclic. We say that a dependency structure $\mathcal{G}$ is *acyclic* relative to a relational skeleton σ_r if the directed graph G_{σ_r} is acyclic.

Theorem 2 (Friedman et al., 1999) *Let Π be a PRM with an acyclic instance dependency graph, and let σ_r be a relational skeleton. Then, Eq. (1) defines a coherent distributions over instances that extend σ_r.*

The definition of the instance dependency graph is specific to the particular skeleton at hand: the existence of an edge from $y.B$ to $x.A$ depends on whether $y \in x.\rho$, which in turn depends on the interpretation of the reference slots. Thus, it allows us to determine the coherence of a PRM only relative to a particular relational skeleton. When we are evaluating different possible PRMs as part of our learning algorithm, we want to ensure that the dependency structure $\mathcal{G}$ we choose results in coherent probability models 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 $Y.B \rightarrow X.A$.

Theorem 3 (Friedman et al., 1999) *If the class dependency graph of a PRM Π is acyclic, then the instance dependency graph for any relational skeleton σ_r is also acyclic. Hence, Π defines a legal model for any relational skeleton σ_r.*

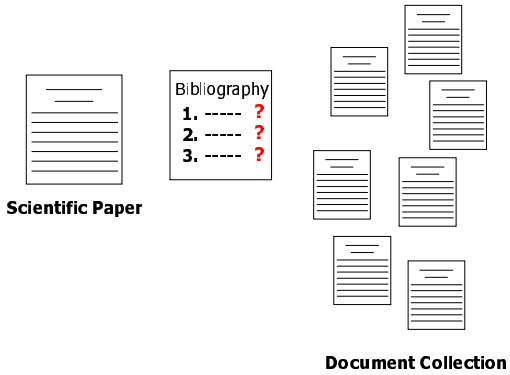


Figure 2: Reference uncertainty in a simple citation domain.

In the model described here, 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 the subsequent sections, we extend our framework to deal with *link uncertainty*, by extending our probabilistic model to include a distribution over the relational structure itself.

3. Reference Uncertainty

In the previous section, we assumed that the relational structure was not a part of our probabilistic model, that it was given as background knowledge in the relational skeleton. But by incorporating the links into the probabilistic model, we can both predict links and more importantly use the links to help us make predictions about other attributes in the model.

Consider a simple citation domain illustrated in Figure 2. Here we have a document collection. Each document has a bibliography that references some of the other documents in the collection. We may know the number of citations made by each document (i.e., it is outside the probabilistic model). By observing the citations that are made, we can use the links to reach conclusions about other attributes in the model. For example, by observing the number of citations to papers of various topics, we may be able to infer something about the topic of the citing paper.

Figure 3(a) shows a simple schema for this domain. We have two classes, Paper and Cites. The Paper class has information about the topic of the paper and the words contained in the paper. For now, we simply have an attribute for each word that is *true* if the word occurs in the page and *false* otherwise. The Cites class represents the citation of one paper, the *Cited* paper, by another paper, the *Citing* paper. (In the figure, for readability, we show the Paper class twice.) In this model, we assume that the set of objects is pre-specified, but relations among them, i.e., reference slots, are subject to probabilistic 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, the object skeleton specifies the objects in class Paper and the objects in class Cites, but the reference slots of the Cites relation, Cites.*Cited* and Cites.*Citing* are unspecified. In other words, the probabilistic model does not provide a model of the total number of citation links, but only a distribution over their "endpoints". Figure 3 shows an object skeleton for the citation domain.

(a) (b)

Figure 3: (a) A relational schema for the citation domain. (b) An object skeleton for the citation domain.

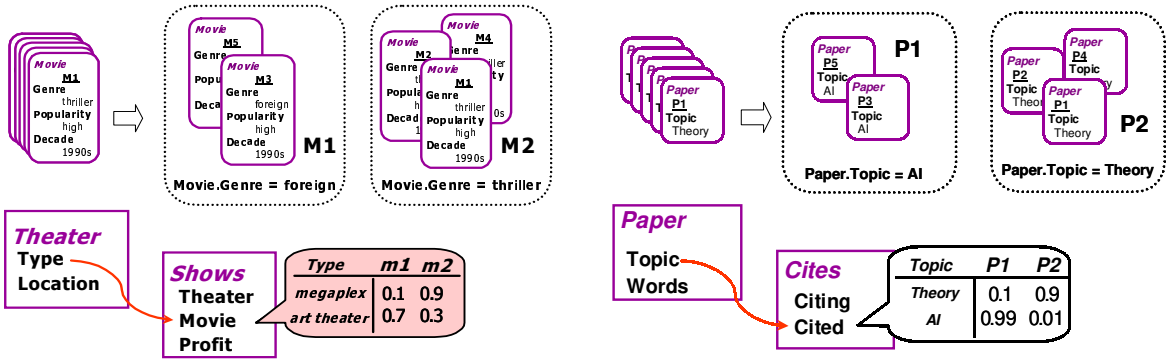
3.1 Probabilistic model

In the case of reference uncertainty, 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. For example for Cites.*Cited*, we must specify a distribution over the objects in class Paper.

A naive approach is to simply have the PRM specify a probability distribution directly over the objects $\sigma_o(Y)$ in Y. For example for Cites.*Cited*, we would have to specify a distribution over the primary keys of Paper. This approach has two major flaws. Most obviously, this distribution would require a parameter for each object in Y, leading to a very large number of parameters. This is a problem both from a computational perspective — the model becomes very large, and from a statistical perspective — we often would not have enough data to make robust estimates for the parameters. 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.

In order to achieve a general and compact representation, we use the *attributes* of Y to define the probability distribution. In this model, we partition the class Y into subsets labeled $\psi_1, \ldots, \psi_m$ according to the values of some of its attributes, and specify a probability for choosing each partition, i.e., a distribution over the partitions. We then select an object within that partition uniformly.

For example, consider a description of movie theater showings as in Figure 4(a). For the foreign key Shows.*Movie*, we can partition the class Movie by *Genre*, indicating that a movie theater first selects the genre of movie it wants to show, and then selects uniformly among the movies with the selected genre. For example, a movie theater may be much more likely to show a movie which is a thriller in comparison to a foreign movie. Having selected, for example, to show a thriller, the theater then selects the actual movie to show uniformly from within the set of thrillers. In addition, just as in the case of descriptive attributes, the partition choice can depend on other attributes in our model. Thus, the selector attribute can have parents. As illustrated in the figure, the choice of movie genre might depend on the type of theater. Consider another example, in our citation domain. As shown in Figure 4(b), we can partition the class Paper by *Topic*, indicating that the topic of a



(a) (b)

Figure 4: (a) An example of reference uncertainty for a movie theater's showings. (b) A simple example of reference uncertainty in the citation domain

citing paper determines the topics of the papers it cites; and then the cited paper is chosen uniformly among the papers with the selected topic.

We make this intuition precise by defining, for each slot ρ, a *partition function* Ψ_ρ. We place several restrictions on the partition function which are captured in the following definition:

Definition 4 *Let $X.\rho$ be a reference slot with domain Y. Let $\Psi_\rho : Y \rightarrow \mathrm{Dom}[\Psi_\rho]$ be a function where $\mathrm{Dom}[\Psi_\rho]$ is a finite set of labels. We say that Ψ_ρ is a* partition function *for ρ if there is a subset of the attributes of Y, $\mathcal{P}[\rho] \subseteq \mathcal{A}(Y)$, such that for any $y \in Y$ and any $y' \in Y$, if the values of the attributes $\mathcal{P}[\rho]$ of y and y' are the same, i.e., for each $A \in \mathcal{P}[\rho]$, $y.A = y'.A$, then $\Psi_\rho(y) = \Psi_\rho(y')$. We refer to $\mathcal{P}[\rho]$ as the* partition attributes *for ρ.* ∎

Thus, the values of the partition attributes are all that is required to determine the partition to which an object belongs.

In our first example, $\Psi_{\mathsf{Shows}.Movie} : \mathsf{Movie} \rightarrow \{Foreign, Thriller\}$ and the partition attribute is $\mathcal{P}[\mathsf{Shows}.Movie] = \{Genre\}$. In the second example, $\Psi_{\mathsf{Cites}.Cited} : \mathsf{Paper} \rightarrow \{AI, Theory\}$ and the partition attribute is $\mathcal{P}[\mathsf{Cites}.Cited] = \{Topic\}$.

There are a number of natural methods for specifying the partition function. It can be defined simply by having one partition for each possible combination of values of the partition attributes, i.e., one partition for each value in the cross product of the partition attribute values. Our examples above take this approach. In both cases, there is only a single partition attribute, so specifying the partition function in this manner is not too unwieldy, but for larger collections of partition attributes or for partition attributes with large domains, this method for defining the partitioning function may be problematic. A more flexible and scalable approach is to define the partition function using a decision tree built over the partition attributes. In this case, there is one partition for each of the leaves in the decision tree.

Each possible value ψ determines a subset of Y from which the value of ρ (the referent) will be selected. For a particular instantiation $\mathcal{I}$ of the database, we use $\mathcal{I}(Y_\psi)$ to represent the set of objects in $\mathcal{I}(Y)$ that fall into the partition ψ.

We now represent a probabilistic model over the values of ρ by specifying a distribution over possible partitions, which encodes how likely the reference value of ρ is to fall into one partition versus another. We formalize our intuition above by introducing a *selector attribute* S_ρ, whose domain is $\mathrm{Dom}[\Psi_\rho]$. The specification of the probabilistic model for the selector attribute S_ρ is the same as that of any other attribute: it has a set of parents and a CPD. In our earlier example, the CPD of $\mathsf{Show}.S_{Movie}$ might have as a parent $\mathsf{Theater}.Type$. For each instantiation of the parents, we have a distribution over $\mathrm{Dom}[S_\rho]$.[3] The choice of value for S_ρ determines the partition Y_ψ from which the reference value of ρ is chosen; the choice of reference value for ρ is uniformly distributed within this set.

Definition 5 *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 partition function Ψ_ρ with a set of partition attributes $\mathcal{P}[\rho] \subseteq \mathcal{A}(Y)$;*
- *a new selector attribute S_ρ within X which takes on values in the range of Ψ_ρ;*
- *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:

$$P(\mathcal{I} \mid \sigma_o, \Pi) = \prod_{X \in \mathcal{X}} \prod_{x \in \sigma_o(X)} \prod_{A \in \mathcal{A}(X)} P(x.A \mid \mathrm{Pa}(x.A)) \prod_{\rho \in \mathcal{R}(X), y = x.\rho} \frac{P(x.S_\rho = \psi[y] \mid \mathrm{Pa}(x.S_\rho))}{|\mathcal{I}(Y_{\psi[y]})|} \tag{2}$$

where $\psi[y]$ refers to $\Psi_\rho(y)$ — the partition that the partition function assigns y. Note that the last term in Eq. (2) depends on $\mathcal{I}$ in three ways: the interpretation of $x.\rho = y$, the values of the attributes $\mathcal{P}[\rho]$ within the object y, and the size of $Y_{\psi[y]}$.

There is one small problem with this definition: the probability is not well-defined if there are no objects in a partition, i.e., $|\mathcal{I}(Y_\psi)| = 0$. In this case, we will perform a renormalization of the distribution for the selector attribute, removing all probability mass from the empty partition. We then treat this term, $\frac{P(x.S_\rho = \psi[x.\rho] \mid \mathrm{Pa}(x.S_\rho)}{)}|Y_\psi|$, as 0 in the above product. In other words, an instantiation where $x.S_\rho = \psi$ and ψ is an empty partition necessarily has probability zero.[4]

3. In the current work, we treat this distribution as a simple multinomial distribution over this value space. In general, however, we can represent such a distribution more compactly, e.g., using a Bayesian network. For example, the genre of movies shown by a movie theater might depend on its type (as above). However, the language of the movie can depend on the location of the theater. Thus, the partition will be defined by $\mathcal{P}(\mathsf{Show}.Movie) = \{\mathsf{Movie}.Genre, \mathsf{Movie}.Language\}$, and its parents would be $\mathsf{Theater}.Type$ and $\mathsf{Theater}.Location$. We can represent this conditional distribution more compactly by introducing a separate variable $S_{\mathsf{Movie}.Genre}$, with a parent $\mathsf{Theater}.Type$, and another $S_{\mathsf{Movie}.Language}$, with a parent $\mathsf{Theater}.Location$.
4. While an important technical detail, in practice, for the majority of our experimental domains, we did not encounter empty partitions.

3.2 Coherence of the Probabilistic Model

As in the case of PRMs with attribute uncertainty, we must be careful to guarantee that our probability distribution is in fact coherent. In this case, the object skeleton does not specify which objects are related to which, and therefore the mapping of formal to actual parents depends on probabilistic choices made in the model. The associated ground Bayesian network will therefore be cumbersome and not particularly intuitive. We define our coherence constraints using an instance dependency graph, relative to our PRM and object skeleton.

Definition 6 *The* instance dependency graph *for a PRM* Π *and an object skeleton* σ_o *is a graph* G_{σ_o} *with the nodes and edges described below. For each class* X *and each* $x \in \sigma_o(X)$*, we have the following nodes:*

- *a node* $x.A$ *for every descriptive attribute* $X.A$*;*
- *a node* $x.\rho$ *and a node* $x.S_\rho$*, for every reference slot* $X.\rho$*.*

The dependency graph contains five types of edges:

- **Type I edges:** *Consider any attribute (descriptive or selector)* $X.A$ *and formal parent* $X.B$*. We define an edge* $x.B \rightarrow x.A$*, for every* $x \in \sigma_o(X)$*.*
- **Type II edges:** *Consider any attribute (descriptive or selector)* $X.A$ *and formal parent* $X.\tau.B$ *where* $\mathrm{Dom}[X.\tau] = Y$*. We define an edge* $y.B \rightarrow x.A$*, for every* $x \in \sigma_o(X)$ *and* $y \in \sigma_o(Y)$*.*
- **Type III edges:** *Consider any attribute* $X.A$ *and formal parent* $X.\tau.B$*, where* $\tau = \rho_1, \ldots, \rho_k$*, and* $\mathrm{Dom}[\rho_i] = X_i$*. We define an edge* $x.\rho_1 \rightarrow x.A$*, for every* $x \in \sigma_o(X)$*. In addition, for* $i > 1$*, we add an edge* $x_i.\rho_i \rightarrow x.A$ *for for every* $x_i \in \sigma_o(X_i)$ *and for every* $x \in \sigma_o(X)$*.*
- **Type IV edges:** *Consider any slot* $X.\rho$ *and partition attribute* $Y.B \in \mathcal{P}[\rho]$ *for* $Y = \mathrm{Range}[\rho]$*. We define an edge* $y.B \rightarrow x.S_\rho$*, for every* $x \in \sigma_o(X)$ *and* $y \in \sigma_o(Y)$*.*
- **Type V edges:** *Consider any slot* $X.\rho$*. We define an edge* $x.S_\rho \rightarrow x.\rho$*, for every* $x \in \sigma_o(X)$*.*

We say that a dependency structure $\mathcal{G}$ *is* acyclic *relative to an object skeleton* σ_o *if the directed graph* G_{σ_o} *is acyclic.* ∎

Intuitively, type I edges correspond to intra-object dependencies and type II edges to inter-object dependencies. These are the same edges that we had in the dependency graph for regular PRMs, except that they also apply to selector attributes. Moreover, there is an important difference in our treatment of type II edges. In this case, the skeleton does not specify the value of $x.\rho$, and hence we cannot determine from the skeleton on which object y the attribute $x.A$ actually depends. Therefore, our instance dependency graph must include an edge from every attribute $y.B$.

Type III edges represent the fact that the actual choice of parent for $x.A$ depends on the value of the slots used to define it. When the parent is defined via a slot-chain, the actual choice depends on the values of all the slots along the chain. Since we cannot determine the particular object from the skeleton, we must include an edge from every slot $x_i.\rho_i$ potentially included in the chain.

Type V edges represent the dependency of a slot on the attributes defining the associated partition. To see why this dependence is required, we observe that our choice of reference value for $x.\rho$ depends on the values of the partition attributes $\mathcal{P}[x.\rho]$ of all of the different objects y in Y. Thus,

these attributes must be determined before $x.\rho$ is determined. Finally, type V edges represent the fact that the actual choice of parent for $x.A$ depends on the value of the selector attributes for the slots used to define it. In our example, as $\mathcal{P}[\mathsf{Shows}.\mathit{Movie}] = \{\mathsf{Movie}.\mathit{Genre}\}$, the genres of all movies must be determined before we can select the value of the reference slot $\mathsf{Shows}.\mathit{Movie}$.

Based on this definition, we can specify conditions under which Eq. (2) specifies a coherent probability distribution.

Theorem 7 *Let Π be a PRM with reference uncertainty whose dependency structure $\mathcal{G}$ is acyclic relative to an object skeleton σ_o. Then Π and σ_o define a coherent probability distribution over instantiations $\mathcal{I}$ that extend σ_o via Eq. (2).*

Proof: The probability of an instantiation $\mathcal{I}$ is the joint distribution over a set of random variables defined via the object skeleton. We have two types of variables:

- We have one random variable $x.A$ for each $x \in \sigma_o(X)$ and each $A \in \mathcal{A}(X)$. Note that this also includes variables of the form $x.S_\rho$ that correspond to selector variables.
- We have one random variable $x.\rho$ for each reference slot $\rho \in \mathcal{R}(X)$ and $x \in \sigma_o(X)$. This variable denotes the actual object in $\sigma_o(\mathrm{Range}[\rho])$ that the slot points to.

Let $V_1, \ldots, V_N$ be the entire set of variables defined by the object skeleton. Clearly, there is a one-to-one mapping between joint assignments to these variables and instantiations $\mathcal{I}$ of the PRM that are consistent with σ_o. Because the instance dependency graph is acyclic, we can assume without loss of generality that the ordering $V_1, \ldots, V_N$ is a topological sort of the instance dependency graph; thus, if $V_i = x.A$, then all ancestors of V_i in the instance dependency graph appear before it in the ordering.

Our proof will use the following argument. Assume that we can construct a non-negative function $f(V_i, V_{i-1}, \ldots, V_1)$ which has the form of a conditional distribution $P(V_i \mid V_1, \ldots, V_{i-1})$, i.e., for any assignment $V_1, \ldots, V_{i-1}$ to $V_1, \ldots, V_{i-1}$, we have that

$$\sum_{V_i} f(V_i, V_{i-1}, \ldots, V_1) = 1. \tag{3}$$

Then, by the chain rule for probabilities, we can define

$$\begin{aligned} P(V_1, \ldots, V_N) &= \prod_{i=1}^{N} f(V_i, V_{i-1}, \ldots, V_1) \\ &= \prod_{i=1}^{N} P(V_i \mid V_1, \ldots, V_{i-1}). \end{aligned}$$

If f satisfies Eq. (3), then P is a well-defined joint distribution.

All that remains is to define the function f in a way that it satisfies Eq. (3). Specifically, the function f will be defined via Eq. (2). We consider the two types of variables in our distribution.

Suppose that V_i is of the form $x.A$, and consider any parent of $X.A$. There are two cases:

- If the parent is of the form $X.B$, then by the existence of type I edges, we have that $x.B$ precedes $x.A$ in G_{σ_o}. Hence, the variable $x.B$ precedes V_i.

- If the parent is of the form $X.\tau.A$, then by the existence of type III edges, for each ρ_i along τ $x_i.\rho_i$ precedes $x.A$ in G_{σ_o} and hence, by the existence of type V edges all of the $x.S_{\rho_i}$ also precede $x.A$. Furthermore, by the existence of type II edges, for any $y \in x.\tau$, we have that $y.B$ precedes $x.A$.

In both cases, we can define

$$P(V_i \mid V_{i-1}, \ldots, V_1) = P(x.A \mid \text{Pa}(x.A))$$

as in Eq. (2). As the right-hand-side is simply a CPD in the PRM, it specifies a well-defined conditional distribution, as required.

Now, suppose that V_i is of the form $x.\rho$. In this case, the conditional probability depends on $x.S_\rho$, and the value of the partition attributes of all objects in $\sigma_o(\text{Range}[\rho])$. By the existence of type V edges, $x.S_\rho$ precedes $x.\rho$. Furthermore, by the existence of type IV edges, we have that $y.B$ for every $Y.B \in \mathcal{P}[\rho]$ and $y \in \sigma_o(\text{Range}[\rho])$. Consequently, the assignment to $V_1, \ldots, V_{i-1}$ determines the number of objects in each partition of values of $\mathcal{P}[\rho]$ and hence the set $\mathcal{I}(Y_\psi)$ for every ψ.

Finally, we set

$$P(V_i = y \mid V_{i-1}, \ldots, V_1) = \begin{cases} \frac{1}{|\mathcal{I}(Y_{\psi[y]})|} & \text{if } \psi[y] = x.S_\rho \\ 0 & \text{otherwise} \end{cases}$$

which is a well defined distribution on the objects in $\sigma_o(Y)$. ∎

This theorem is limited in that it is very specific to the constraints of a given object skeleton. As in the case of PRMs without relational uncertainty, we want to learn a model in one setting, and be assured that it will be acyclic for any skeleton we might encounter. We accomplish this goal by extending our definition of class dependency graph. We do so by extending the class dependency graph to contain edges that correspond to the edges we defined in the instance dependency graph.

Definition 8 *The* class dependency graph G_Π *for a PRM with reference uncertainty* Π *has a node for each descriptive or selector attribute* $X.A$ *and each reference slot* $X.\rho$*, and the following edges:*

- **Type I edges:** *For any attribute* $X.A$ *and formal parent* $X.B$*, we have an edge* $X.B \rightarrow X.A$*.*
- **Type II edges:** *For any attribute* $X.A$ *and formal parent* $X.\rho.B$ *where* $\text{Range}[\rho] = Y$*, we have an edge* $Y.B \rightarrow X.A$*.*
- **Type III edges:** *For any attribute* $X.A$ *and formal parent* $Y.\tau.B$*, where* $\tau = \rho_1, \ldots, \rho_k$*, and* $\text{Dom}[\rho_i] = X_i$*, we define an edge* $X.\rho_1 \rightarrow X.A$*. In addition, each* $i > 1$*, we add an edge* $X.\rho_i \rightarrow X.A$*.*
- **Type IV edges:** *For any slot* $X.\rho$ *and partition attribute* $Y.B$ *for* $Y = \text{Range}[\rho]$*, we have an edge* $Y.B \rightarrow X.S_\rho$*.*
- **Type V edges:** *For any slot* $X.\rho$*, we have an edge* $X.S_\rho \rightarrow X.\rho$*.*

Figure 5 shows the class dependency graph for our extended movie example.

It is now easy to show that if this class dependency graph is acyclic, then the instance dependency graph is acyclic.

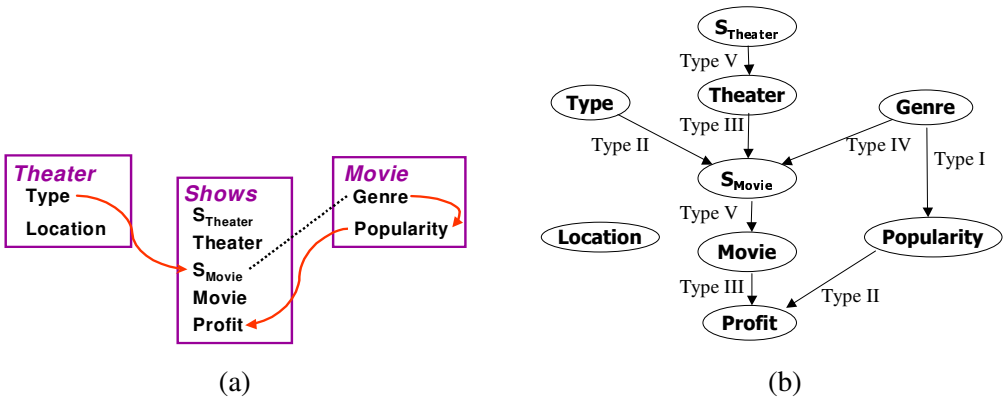


Figure 5: (a) A PRM for the movie theater example. The partition attributes are indicated using dashed lines. (b) The dependency graph for the movie theater example. The different edge types are labeled.

Lemma 9 *If the class dependency graph is acyclic for a PRM with reference uncertainty* Π*, then for* any *object skeleton* σ_o*, the instance dependency graph is acyclic.*

Proof: Assume by contradiction that there is a cycle

$$x_1.V_1 \rightarrow x_2.V_2 \cdots x_k.V_k \rightarrow x_1.V_1.$$

Then, because each of these object edges corresponds to an edge in the class dependency graph, we have the following cycle in the class dependency graph:

$$X_1.V_1 \rightarrow X_2.V_2 \cdots X_k.V_k \rightarrow X_1.V_1.$$

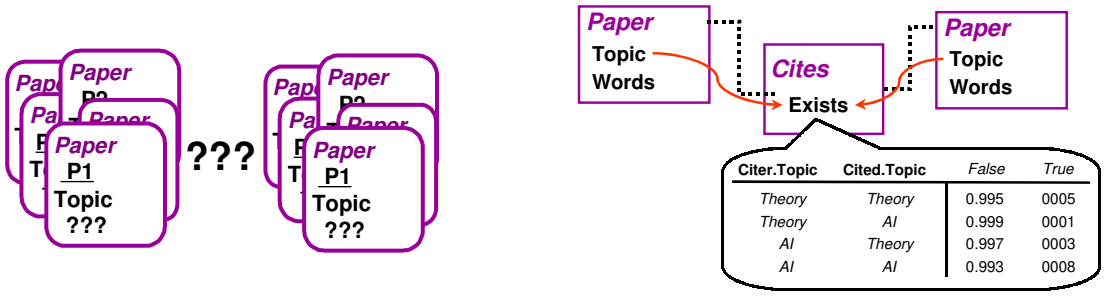
This contradicts our hypothesis that the class dependency graph is acyclic. ∎

The following corollary follows immediately:

Corollary 10 *Let* Π *be a PRM with reference uncertainty whose class dependency structure* $\mathcal{G}$ *is acyclic. For for any object skeleton* σ_o*,* Π *and* σ_o *define a coherent probability distribution over instantiations* $\mathcal{I}$ *that extend* σ_o *via Eq. (2).*

4. Existence Uncertainty

The second form of structural uncertainty we introduce is called *existence uncertainty*. In this case, we make no assumptions about the number of links that exist. The number of links that exist and the identity of the links are all part of the probabilistic model and can be used to make inferences about other attributes in our model. In our citation example above, 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. Unlike the reference uncertainty model of the previous section, we do not assume that the total number of citations is fixed, but rather that each potential citation can be present or absent.



(a) (b)

Figure 6: (a) An entity skeleton for the citation domain. (b) A CPD for the *Exists* attribute of Cites.

4.1 Semantics of relational model

The object skeleton used for reference uncertainty assumes that the number of objects in each relation 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. Existence uncertainty assumes even less background information than specified by the object skeleton. Specifically, we assume that the number of relationship objects is not fixed in advance.

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. Figure 6(a) shows an entity skeleton for the citation example. Our basic approach is to allow other objects within the model — those in the relationship classes — to be *undetermined*, i.e., their existence can be uncertain. In other words, we introduce into the model all of the objects that can *potentially* exist in it; with each of them, we associate a special binary variable that tells us whether the object actually exists or not. We call entity classes *determined* and relationship classes *undetermined*.

To specify the set of potential objects, 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 has the two reference slots *Citing* and *Cited*. Thus the potential domain of the Cites class in a given instantiation $\mathcal{I}$ is $\mathcal{I}(\mathsf{Paper}) \times \mathcal{I}(\mathsf{Paper})$. Each "potential" object x in this class has to form $\mathsf{Cite}[y_1, y_2]$. Each such object is associated with a binary attribute $x.E$ that specifies whether paper y_1 did or did not cite paper y_2.

Definition 11 *Consider a schema with determined and undetermined classes, and let σ_e be an entity skeleton over this schema. We define the* induced *relational skeleton, $\sigma_r[\sigma_e]$, to be the relational skeleton that contains the following objects:*

- *If X is a determined class, then $\sigma_r[\sigma_e](X) = \sigma_e(X)$.*
- *Let X be an undetermined class with reference slots $\rho_1, \ldots, \rho_k$ whose range types are $Y_1, \ldots, Y_k$ respectively. Then $\sigma_r[\sigma_e](X)$ contains an object $X[y_1, \ldots, y_k]$ for all tuples $\langle y_1, \ldots, y_k \rangle \in \sigma_r[\sigma_e](Y_1) \times \cdots \times \sigma_r[\sigma_e](Y_k)$.*

The relations in $\sigma_r[\sigma_e]$ are defined in the obvious way: Slots of objects of determined classes are taken from the entity skeleton. Slots of objects of undetermined classes are induced from the object definition: $X[y_1, \ldots, y_k].\rho_i$ is y_i. ∎

To ensure that the semantics of schemas with undetermined classes is well-defined, we need a few tools. Specifically, we need to ensure that the set of potential objects is well defined and finite. It is clear that if we allow cyclic references (e.g., an undetermined class with a reference to itself), then the set of potential objects is not finite. To avoid such situations, we need to put some requirements on the schema.

Definition 12 *A set of classes $\mathcal{X}$ is* stratified *if there exists a partial ordering over the classes $\prec$ such that for any reference slot $X.\rho$ with range type Y, $Y \prec X$.* ∎

Lemma 13 *If the set of undetermined classes in a schema is stratified, then given any entity skeleton σ_e the number of potential objects in any undetermined class is finite.*

Proof: We prove this by induction on the stratification level of the class X. Let $\rho_1, \ldots, \rho_k$ be the set of reference slots of X, and let $Y_i = \mathrm{Range}[\rho_i]$. Then $\mathcal{I}(X) = \mathcal{I}(Y_1) \times \cdots \mathcal{I}(Y_k)$. If X is the first level in the stratification, it cannot refer to any undetermined classes. Thus, the number of potential objects in $\mathcal{I}(X)$ is simply the product of the number of objects in each determined class Y_i as specified in the entity skeleton σ_e. Each term is finite, so the product of the terms will be finite.

Next, assume by induction that all of the classes at stratification levels less than i are finite. If X is at level i in the stratification, the constraints imposed by the stratification constraints imply that the range type of all of its reference slots are at stratification levels $< i$. By our induction hypothesis, each of these classes is finite. Hence, the cross product of the classes of the reference slots is finite and the number of potential objects in X is finite. ∎

As discussed, each undetermined 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.

For a PRM with stratified undetermined classes, we define an instantiation to be an assignment of values to the attributes, including the *Exists* attribute, of *all* potential objects.

4.2 Probabilistic model

We now specify the probabilistic model defined by the PRM. By treating the *Exists* attributes as standard descriptive attributes, we can essentially build our definition directly on top of the definition of standard PRMs.

Specifically, 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 has an associated CPD. Figure 6(b) illustrates a CPD for the Cites.*Exists* attribute. In this example, the existence of a citation depends on the topic of the citing paper and the topic of the cited paper; e.g., it is more likely that citations will exist between papers with the same topic.[5].

5. This is similar to the 'encyclopedic' links discussed by (Ghani et al., 2001). Our exists models can capture each of the types of links (encyclopedic, co-referencing, and partial co-referencing) defined in (Ghani et al., 2001). Moreover our exists models are much more general, and can capture much richer patterns in the existence of links.

Using the induced relational skeleton, treating the existence events as descriptive attributes, we have set things up so that Eq. (1) applies with minor changes. There are two minor changes to the definition of the distribution:

- We want to enforce that $x.E = \mathit{false}$ if $x.\rho.E = \mathit{false}$ for one of the slots ρ of X. Suppose that X has the slots $\rho_1, \ldots, \rho_k$, we define the *effective* CPD for $X.E$ as follows. Let $\text{Pa}^*(X.E) = \text{Pa}(X.E) \cup \{X.\rho_1.E, \ldots, X.\rho_k.E\}$, and define

$$P^*(X.E \mid \text{Pa}^*(X.E)) = \begin{cases} P(X.E \mid \text{Pa}(X.E)) & \text{if } X.\rho_i.E = \mathit{true}, \forall i = 1, \ldots, k \\ 0 & \text{otherwise} \end{cases}$$

- We want to "decouple" the attributes of non-existent objects from the rest of the PRM. Thus, if $X.A$ is a descriptive attribute, we define $\text{Pa}^*(X.A) = \text{Pa}(X.A) \cup \{X.E\}$, and

$$P^*(X.A \mid \text{Pa}^*(X.A)) = \begin{cases} P(X.A \mid \text{Pa}(X.A)) & \text{if } X.E = \mathit{true} \\ \frac{1}{|V(X.A)|} & \text{otherwise} \end{cases}$$

It is easy to verify that in both cases $P^*(X.A \mid \text{Pa}^*(X.A))$ is a legal conditional distribution.

In effect, these constraints specify a new PRM Π^*, in which we treat $X.E$ as a standard descriptive attribute. For each attribute (including the *Exists* attribute), we define the parents of $X.A$ in Π^* to be $\text{Pa}^*(X.A)$ and the associated CPD to be $P^*(X.A \mid \text{Pa}^*(X.A)$.

Given an entity skeleton σ_e, a PRM with exists uncertainty Π specifies a distribution over a set of instantiations $\mathcal{I}$ consistent with $\sigma_r[\sigma_e]$:

$$P(\mathcal{I} \mid \sigma_e, \Pi) = P(\mathcal{I} \mid \sigma_r[\sigma_e], \Pi^*) = \prod_{X \in \mathcal{X}} \prod_{x \in \sigma_r[\sigma_e](X)} \prod_{A \in \mathcal{A}(x)} P^*(x.A \mid \text{Pa}^*(x.A)) \qquad (4)$$

We can similarly define the instance dependency graph and the class dependency graph for a PRM Π with existence uncertainty using the corresponding notions for the standard PRM Π^*. As there, we require that the class dependency graph G_{Π^*} is acyclic. One immediate consequence of this requirement is that the schema is stratified.

Lemma 14 *If the class dependency graph G_{Π^*} is acyclic, then there is a stratification of the undetermined classes.*

Proof: The stratification is given by an ordering of *Exists* attributes consistent with the class dependency graph. Because the class dependency graph has an edge from $Y.E$ to $X.E$ for every slot $\rho \in \mathcal{R}(X)$ whose range type is Y, Y will precede X in the constructed ordering. Hence it is a stratification ordering. ∎

Furthermore, based on this definition, we can now easily prove the following result:

Theorem 15 *Let Π be a PRM with existence uncertainty and an acyclic class dependency graph. Let σ_e be an entity skeleton. Then Eq. (4) defines a coherent distribution on all instantiations $\mathcal{I}$ of the induced relational skeleton $\sigma_r[\sigma_e]$.*

Proof By Lemma 14 and Lemma 13 we have that $\sigma_r[\sigma_e]$ is a well-defined relational skeleton. Using the assumption that G_{Π^*} is acyclic, we can apply Theorem 2 to Π^* and conclude that Π^* defines a coherent distribution over instances to σ_r, and hence so does Π. ■

One potential shortcoming of our semantics is that it defines probabilities over instances that include assignment of descriptive attributes to non-existent objects. This potentially presents a problem. An actual instantiation (i.e., a database) will contain value assignments only for the descriptive attributes of existing objects. This suggests that in order to compute the likelihood of a database we need to sum over all possible values of descriptive attributes of potential objects that do not exist. (As we shall see, such likelihood computations are integral part of our learning procedure.) Fortunately, we can easily see that the definition of P^* ensures that if $x.E = 0$, then variables of the form $x.A$ are *independent* of all other variables in the instance. Thus, we can ignore such descriptive attributes when we compute the likelihood of a database.

The situation with *Exists* attribute is somewhat more complex. When we observe a database, we also observe that many potential objects do not exist. The non-existence of these objects can provide information about other attributes in the model, which is taken into consideration by the correlations between them and the *Exists* attributes in the PRM. At first glance, this idea presents computational difficulties, as there can be a very large number of non-existent objects. However, we note that the definition of P^* is such that we need to compute $P^*(x.E = \mathit{false} \mid \mathrm{Pa}^*(x.E))$ only for objects whose slots refer to existing objects, thereby bounding the number of non-existent objects we have to consider.

5. 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 two 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 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 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 and 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 $\mathcal{P}[\mathsf{Appearance}.\mathit{HasWord}] = \mathsf{Words}.\mathit{Word}$, 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 and 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; for example they can also depend on other attributes, such as the research area of the document's author.

6. Learning PRMs with Structural Uncertainty

In this section, we briefly review the learning algorithm for the basic PRM framework in Friedman et al. (1999) and describe the modifications needed to handle the two extensions of PRMs proposed in the previous sections. Our aim is to *learn* such models from data: given a schema and an instance, construct a PRM that captures the dependencies between objects in the schema. We stress that basic PRMs and both proposed 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.

6.1 Learning basic PRMs

We separate the learning problem into two tasks: 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 PRM Π given an instantiation $\mathcal{I}$. Using Bayes rule we have that $P(\Pi \mid \mathcal{I}) \propto P(\mathcal{I} \mid \Pi)P(\Pi)$. This score is composed of two main parts: the prior probability of Π, and the probability of the instantiation assuming the PRM is Π. By making fairly reasonable assumptions about the prior probability of structures and parameters, this term can be *decomposed* into a product of terms Friedman et al. (1999). As in Bayesian network learning, each term in the decomposed form of the score 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 $\mathrm{Pa}(x.A) = \mathbf{u}$.

Model Search To find a high-scoring structure in basic PRM framework, we use a simple search procedure that considers operators such as adding, deleting, or reversing edges in the dependency model Π. The procedure performs greedy hill-climbing search, using the Bayesian score to evaluate structures. As in Bayesian network search, we can take advantage of score decomposibility to perform the hill-climbing efficiently: after adding or deleting a parent of an attribute, the only steps that need to be re-scored are other edges that add/delete parents for this attribute.

6.2 Learning with reference uncertainty

The extension to scoring required to deal with reference uncertainty is not a difficult one. Once we fix the partitions defined by the attributes $\mathcal{P}[\rho]$, 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).

The extension to search the model space for incorporating reference uncertainty involves expanding our search operators to allow the addition (and deletion) of attributes to partition definition for each reference slot. 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 $\mathcal{P}[\rho]$. We introduce two new operators **refine** and **abstract**, which modify the partition by adding and deleting attributes from $\mathcal{P}[\rho]$. Initially, $\mathcal{P}[\rho]$ is empty for each ρ. The **refine** operator adds an attribute into $\mathcal{P}[\rho]$; the **abstract** operator deletes one. As mentioned earlier, we can define the partition simply by looking at the cross product of the values for each of the partition attributes, or using a decision tree. In the case of a decision tree, **refine** adds a split to one of the leaves and **abstract** removes a split. 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.

6.3 Learning with Existence Uncertainty

The extension of the Bayesian score to PRMs with existence uncertainty is straightforward; the exists attribute is simply a new descriptive attribute. The only new issue is how to compute sufficient statistics that include existence attributes $x.E$ without explicitly enumerating all the non-existent entities. We perform this computation by counting, for each possible instantiation of $\text{Pa}(X.E)$, the number of potential objects with that instantiation, and subtracting the actual number of objects x with that parent instantiation. Let $\mathbf{u}$ be a particular instantiation of $\text{Pa}(X.E)$. To compute $C_{X.E}[\textit{true}, \mathbf{u}]$, we can use standard database query to compute how many objects $x \in \sigma(X)$ have $\text{Pa}(x.E) = \mathbf{u}$. To compute $C_{X.E}[\textit{false}, \mathbf{u}]$, we need to compute the number of *potential* entities. We can do this without explicitly considering each $(x_1, \ldots, x_k) \in \mathcal{I}(Y_1) \times \cdots \mathcal{I}(Y_k)$ by decomposing the computation as follows: Let ρ be a reference slot of X with $\text{Range}[\rho] = Y$. Let $\text{Pa}_\rho(X.E)$ be the subset of parents of $X.E$ along slot ρ and let $\mathbf{u}_\rho$ be the corresponding instantiation. We count the number of y consistent with $\mathbf{u}_\rho$. If $\text{Pa}_\rho(X.E)$ is empty, this count is simply $|\mathcal{I}(Y)|$. The product

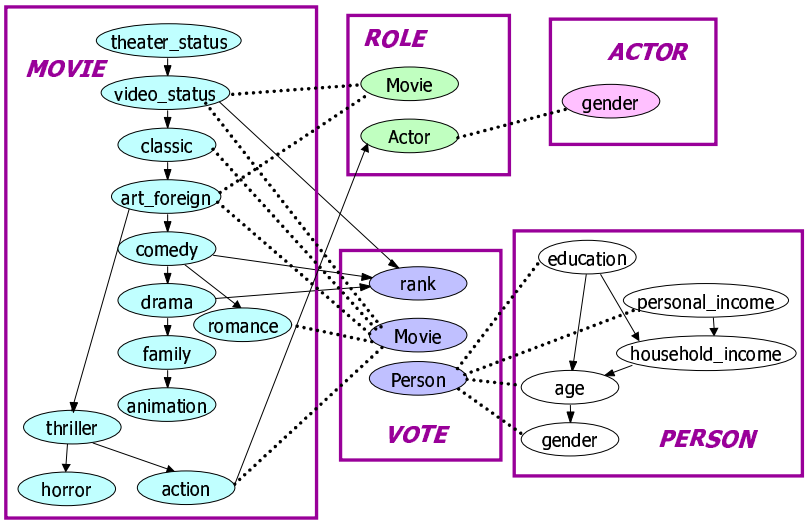


Figure 7: Π_{RU}, the PRM learned using reference uncertainty. The reference slots are Role.*Movie*, Role.*Actor*, Vote.*Movie*, and Vote.*Person*. Dashed lines indicate attributes used in defining the partition.

of these counts is the number of potential entities. To compute $C_{X.E}[\mathit{false}, \mathbf{u}]$, we simply subtract $C_{X.E}[\mathit{true}, \mathbf{u}]$ from this number.

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. Our search algorithm now considers operators that add, delete or reverse edges involving the exist attributes. As usual, we enforce coherence using the class dependency graph. In addition to having an edge from $Y.E$ to $X.E$ for every slot $\rho \in \mathcal{R}(X)$ whose range type is Y, when we add an edge from $Y.B$ to $X.A$, we add an edge from $Y.E$ to $X.E$ and an edge from $Y.E$ to $X.A$.

7. Experimental Results

We evaluated our learning algorithms on several real-world data sets. In this section, we describe the PRM learned for a domain using both of our models for representing structural uncertainty. In each case, we compare against a simple baseline model. Our experiments used the Bayesian score with a uniform Dirichlet parameter prior with equivalent sample size $\alpha = 2$, and a uniform distribution over structures. The partitions are defined using the cross-product of values of the partition attributes; they are not with using decision trees.

7.1 Predictive ability

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 model. Unfortunately, we cannot directly compare the likelihood of the EU and RU models, since the

PRMs involve different sets of probabilistic events and condition on varying amounts of background knowledge. Thus to evaluate our models we compare the PRM with structural uncertainty to a "baseline" model which incorporate link probabilities, but makes the "null" assumption that the link structure is uncorrelated with the descriptive attributes. For reference uncertainty, the baseline has $\mathcal{P}[\rho] = \emptyset$ for each slot. For existence uncertainty, it forces $x.E$ to have no parents in the model.

We evaluated these variants on a dataset that combines information about movies and actors from the Internet Movie Database[6] and information about people's ratings of movies from the Each Movie dataset,[7] 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). For RU this was done by modeling the reference uncertainty of the slots of these objects. For EU this was done by modeling probability of the existence of such objects. We evaluated our methods using ten-fold cross validation. To do this, we partitioned our data into ten subsets. For each each subset, we trained on nine-tenths of the data (the data not included in the subset) and evaluated the log-likelihood of the held-out test subset. We then average the results. In both cases, the model using structural uncertainty significantly outperformed its "baseline" counterpart. For RU, we obtained a log-likelihood of $-149,705$ as compared to $-152,280$ for the baseline model. For EU, we obtained a log-likelihood of $-210,044$ for the EU model, as compared to $-213,798$ for the baseline EU 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 7 shows the RU model learned. In the RU model 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 model 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. Furthermore, the actor reference slot has Movie.*Action* as a parent; the CPD encodes the fact that male actors are more likely to have roles in action movies than female actors.

The EU model learned had an exist attribute for both vote and role. In the model we learned, the existence of a vote depended on the age of the voter and the movie genre, and the existence of a role depended on the gender of the actor and the movie genre.

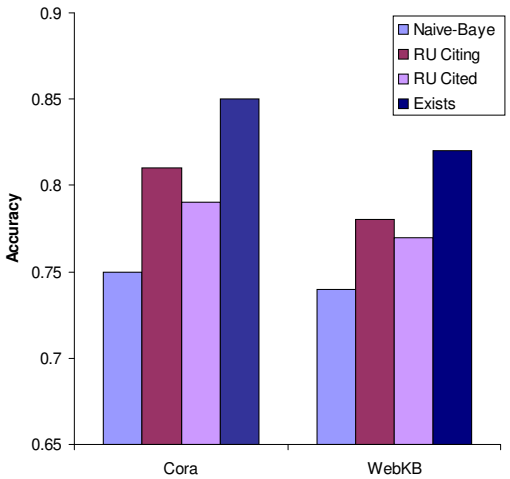
7.2 Classification

While we cannot directly compare the likelihood of the EU and RU models, we can compare the predictive performance of each model. In this set of experiments, we evaluate the predictive accuracy of the different models on two datasets. It is quite interesting to note that we observed that both models of structural uncertainty significantly increase our prediction accuracy.

We considered the conjecture that by modeling link 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

6. ©1990-2000 Internet Movie Database Limited.
7. http://www.research.digital.com/SRC/EachMovie.

Table 1: Prediction accuracy of topic/category attribute of documents in the Cora 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

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 four 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 Cites.*Citing*. The parent of the selector variable is Cites.*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. In this case, an explanation of this phenomenon is that most of the information for this paper 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.

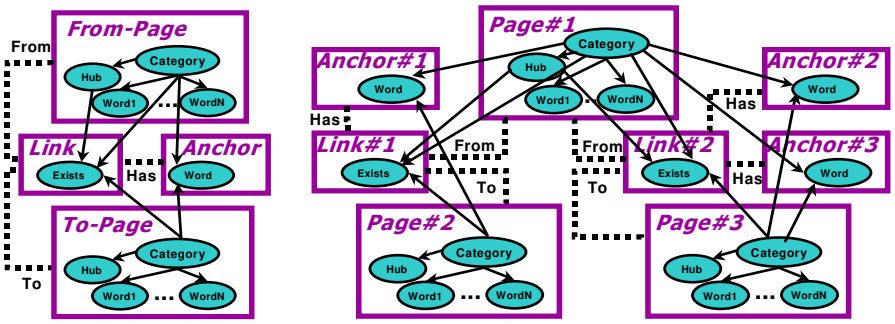
7.3 Collective Classification

In the preceding experiments, the topics of all the linked papers or webpages were observed and we made a prediction for a single unobserved topic. In a more realistic setting, we will have a whole collection of unlabelled instances that are linked together. We can no longer make each prediction in isolation, since for example, the (predicted) topic of one paper influences the (predicted) topics of the papers it cites.

This task of collective classification requires us to reason about the entire collection of instances at once. In our framework, this translates into computing the posterior distribution over the unobserved variables given the data and assigning each unobserved variable its most likely value. This requires inference over the unrolled network defined by instantiating a PRM for a particular document collection. We cannot decompose this task into separate inference tasks over the objects in the model, as they are all correlated. In general, the unrolled network can be fairly complex, involving many documents that are linked in various ways. (In our experiments, the networks involve hundreds of thousands of nodes.) Exact inference over these networks is clearly impractical, so we must resort to approximate inference. Following Taskar et al. (2001), we use belief propagation for the task of inference in PRMs. Belief propagation is a local message passing algorithm introduced by Pearl (1988). Although a very simple approximation, it has lately been shown to be very effective on a wide range of probabilistic models (see, e.g., (Murphy et al., 1999)).

We evaluated several existence uncertainty models for the task of collective classification on the WebKB dataset (Craven et al., 1998). Recall that the WebKB dataset consists of webpages in the Computer Science departments of four schools: Cornell, University of Texas at Austin, University of Washington, and University of Wisconsin.

Figure 8(a) shows a PRM for this domain. For clarity, the Page class is duplicated in the figure, once as From-Page and once as To-Page. Each page has a category attribute representing the type of web page which is one of $\{$course, professor, student, project, other$\}$. The text content of the web page is represented using a set of binary attributes that indicate the presence of different words on the page. After stemming, removing stop words and rare words, the dictionary contains around 800 words. In addition, each page was labelled with a "category hub" attribute, whose domain is $\{$*course, professor, student, project, none*$\}$. A page was labelled with a hub of a particular type (not *none*) if it pointed to many pages of that category. The prevalence of such hubs (or directories) on



(a) (b)

Figure 8: (a) PRM Model for WebKB domain; (b) Fragment of unrolled network for WebKB model.

the web was investigated by Kleinberg (1998), who noted that pages of the same topic or category are often linked to by hub pages. This insight was utilized in the classification algorithm FOIL-HUBS of (Slattery and Mitchell, 2000) for the WebKB dataset. Pages in the training set were hand-labeled with the hub attribute, but the attribute was hidden in the test data. Each school had one hub page of each category, except for Washington which does not have a project hub page and Wisconsin which does not have a faculty web page.

The data set also describes the links between the web pages within a given school. In addition, for each link between pages, the dataset specifies the words (one or more) on the anchor link. There are approximately 100 possible anchor words.

In these experiments, we included only pages that have at least one out link. The number of resulting pages for each school are: Cornell (318), Texas (319), Washington (420), and Wisconsin (465). The number of links for each school are: Cornell (923), Texas (1041), Washington (1534) and Wisconsin (1823).

Given a particular set of hyperlinked pages, the template is instantiated to produce an "unrolled" Bayesian network. Figure 8(b) shows a fragment of such a network for three webpages. The two existing links from page 1 to page 2 and 3 are shown while non-existing links omitted for clarity (however still play a role in the inference). Also shown are the anchor word for link 1 and two anchor words for link 2. Note that during classification, existence of links and anchor words in the links are used as evidence to infer categories of the web pages. Hence, our unrolled Bayes net has active paths between categories of pages through the v-structures at Link.*Exists* and Anchor.*Word*. These active paths capture exactly the pattern of relational inference we set out to model.

We compared the performance of several models on predicting web page categories. In each case, we learned a model from three schools, and tested the performance of the learned model on the remaining school. Our experiments used the Bayesian score with a uniform Dirichlet parameter prior with equivalent sample size $\alpha = 2$.

All models we compared can be viewed as a subset of the model in Figure 8(a). Our baseline is a standard binomial Naive Bayes model that uses only words on the page to predict the category of the page. We evaluated the following set of models:

1. **Naive-Bayes**: Our baseline model.

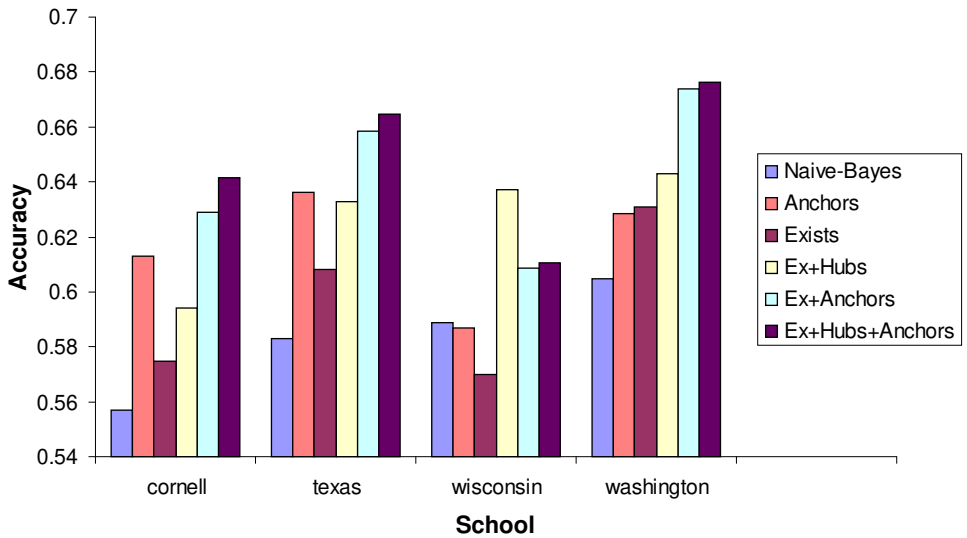


Figure 9: Comparison of accuracy of several models ranging from the simplest model, **Naive-Bayes** to the most complex model, **Ex+Hubs+Anchors**, which incorporates existence uncertainty, hubs and link anchor words. In each case, a model was learned for 3 schools and tested on the remaining school.

2. **Anchors:** This model uses both words on the page and anchor words on the links to predict the category.
3. **Exists**: This model adds structural uncertainty over the link relationship to the simple baseline model; the parents of Link.*Exists* are Link.*From-Page.Category* and Link.*To-Page.Category*.
4. **Ex+Hubs**: This model extends the **Exists** model with Hubs. In the model Link.*Exists* depends on *Link.From-Page.Hub* in addition to the categories of each of the pages.
5. **Ex+Anchors**: This model extends the **Exists** model with includes anchor words (but not hubs).
6. **Ex+Hubs+Anchors**: The final model includes existence uncertainty, hubs and anchor words.

Figure 9 compares the accuracy achieved by the different models on each of the schools. The final model, **Ex+Hubs+Anchors**, which incorporates structural uncertainty, hubs and anchor words, consistently outperforms the **Naive-Bayes** model by a significant amount. In addition, it outperforms any of the simpler variants.

Our algorithm was fairly successful at identifying the hubs in the test set: it recognized 8 out of 14 hubs correctly. However precision was not very high: 38 pages were mislabeled as hubs. The pages mislabeled as hubs often pointed to many pages that had been labeled as Other web pages. However, on further inspection, these hub pages often *were* directories pointing to pages that were likely to be researcher home pages or course home pages and seemed to have been mislabeled in the training set as Other. We investigated how much these misclassifications hurt the performance by

revealing the labels of the hub attribute in the test data. The improvement in classification accuracy of **Ex+Hubs+Anchors** model was roughly 2%.

We also experimented with classification using models where the category of each page has a (multinomial) logistic dependence on the presence of content words, i.e., the arrows from *Category* to the *Word*$_i$'s in Figure 8 are reversed. The logistic CPD is given by

$$P(\mathit{Category} = c \mid \mathit{Word}_1 = w_1, \ldots, \mathit{Word}_n = w_n) \propto \exp\{\theta_{c,0} + \sum \theta_{c,i} w_i\},$$

where each $w_i \in \{0, 1\}$ indicates the presence of word i. We used a zero-mean independent Gaussian prior for the parameters given by

$$P(\theta_{c,i}) \propto \exp\{-\theta_{c,i}^2 / 2\sigma^2\},$$

with $\sigma = 0.3$ and estimated the CPD using the standard conjugate gradient algorithm. The average accuracy (averaged over four leave-one-school out experiments) for the flat logistic model was 82.95%, which is significantly higher than all of the models presented so far. This improvement in performance is not surprising, as discriminatively-trained models such as logistic regression often achieve significantly higher classification accuracies than their generative counterparts.

Unfortunately, adding the exists edges to this model did not improve accuracy, but in fact lowered it to 76.42%. Our conjecture is that the evidence about the category of a page from the large number of exists variables overwhelms the evidence from the content words for the logistic-based models, while the models that extend Naive Bayes are more balanced. The reason is that, in Naive Bayes, the posterior over the category (given the words in the document), is usually very sharply peaked around one value because of the violations of the independence assumption. It takes a significant amount of evidence from the links to change this posterior. The logistic CPD is much better at representing the conditional distribution of category given the words. However, as the posterior is based only on a single CPD, it is not highly skewed, and the large amount of evidence from the exists variables can easily sway it to a new maximum.

To test this conjecture, we tried to reduce the influence of the exists evidence by modifying the CPD of the exists variable to be closer to uniform. One way to accomplish this is to "raise the temperature" or raise the CPD entries to a power between 0 and 1. We discovered that, for values in the range 0.1–0.2, the model gets accuracies that are 2–3% higher than the logistic baseline. This interaction of different sources of influence in a globally consistent model warrants further investigation. One solution to this problem is to discriminatively train the entire model, including the model for the exists variables, as in the recent work of Taskar et al. (2002). As shown in that work, this training allows us to gain improvements in accuracy from discriminative training on the one hand, and from the information flow between related objects on the other.

8. Discussion and Conclusions

In this paper, we proposed 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 within 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. Both of the models we propose are relatively simple. They make certain independence assumptions that often will not hold. Thus the significance of our results is that even these simplistic models of link uncertainty provide us with increased predictive accuracy.

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 problem. 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.

ACKNOWLEDGEMENTS

We gratefully thank Eran Segal for many useful discussions and for his help with the collective classification results. This work was supported by ONR contract N66001-97-C-8554 under DARPA's HPKB program, by Air Force contract F30602-00-2-0598 under DARPA's EELD program, and by the Sloan foundation. Nir Friedman was also supported by Israel Science Foundation grant 244/99 and an Alon Fellowship.

References

S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 307–318, Seattle, Washington, 1998.

D. Cohn and T. Hofmann. The missing link—a probabilistic model of document content and hypertext connectivity. In *Proceedings of Neural Information Processing Systems 13*, pages 430–436, Vancouver, British Columbia, 2001.

M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the world wide web. In *Proceedings of the Fifteenth Conference of the American Association for Artificial Intelligence*, pages 509–516, Madison, Wisconsin, 1998.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1300–1309, Stockholm, Sweden, 1999.

R. Ghani, S. Slattery, and Y. Yang. Hypertext categorization using hyperlink patterns and meta data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, Williamstown, Massachusetts, 2001.

D. Heckerman. A tutorial on learning with Bayesian networks. In M. I. Jordan, editor, *Learning in Graphical Models*. MIT Press, Cambridge, Massachusetts, 1998.

J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677, San Francisco, California, 1998.

D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proceedings of the Fifteenth Conference of the American Association for Artificial Intelligence*, pages 580–587, Madison, Wisconsin, 1998.

Nada Lavrač and Saso Džeroski. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood, New York, New York, 1994.

A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *Workshop on Learning for Text Categorization at the Fifteenth Conference of the American Association for Artificial Intelligence*, Madison, Wisconsin, 1998.

A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

S. Muggleton, editor. *Inductive Logic Programming*. Academic Press, 1992.

K. Murphy, Y. Weiss, and M. Jordan. Loopy belief propagation for approximate inference: an empirical study. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 467–475, Stockholm, Sweden, 1999.

L. Ngo and P. Haddawy. Probabilistic logic programming and bayesian networks. In *Algorithms, Concurrency and Knowledge (Proceedings ACSC95)*, volume 1023 of *Lecture Notes in Computer Science*, pages 286–300. Springer-Verlag, 1995.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Francisco, 1988.

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

A. Popescul, L. Ungar, S. Lawrence, and D. Pennock. Towards structural logistic regression: Combing relational and statistical learning. In *Proc. KDD-2002 Workshop on Multi-Relational Data Mining*. ACM, 2002.

S. Slattery and M. Craven. Combining statistical and relational methods for learning in hypertext domains. In David Page, editor, *Proceedings of ILP-98, 8th International Conference on Inductive Logic Programming*, pages 38–52, Madison, US, 1998. Springer Verlag, Heidelberg, DE.

S. Slattery and T. Mitchell. Discovering test set regularities in relational domains. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 895–902, Stanford, California, 2000.

B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, Edmonton, Canada, 2002.

B. Taskar, E. Segal, and D. Koller. Probabilistic classification and clustering in relational data. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 870–876, Seattle, Washington, 2001.

Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 18(2-3):219–241, 2002.