Modelling Relational Statistics With Bayes Nets

Oliver Schulte, Hassan Khosravi, Arthur E. Kirkpatrick, Tianxiang Gao, and Yuke Zhu

School of Computing Science, Simon Fraser University,
Burnaby, British Columbia, Canada

Abstract. Class-level models capture relational statistics over attributes of objects and their connecting links, answering questions such as "what is the percentage of friendship pairs where both friends are women?" Class-level relationships are important in themselves, and they support applications like policy making, strategic planning, and query optimization. We represent class-level statistics as Parametrized Bayes Nets (PBNs), a first-order logic extension of Bayes nets. Queries about classes require a new semantics for PBNs, as the standard grounding semantics is only appropriate for answering queries about specific ground facts. We propose a *grounding-free* semantics for PBNs that supports class-level queries. The parameters for this semantics can be learned using the recent pseudo-likelihood measure [1] as the objective function. This objective function is maximized by taking the empirical frequencies in the relational data as the parameter settings. We render the computation of these empirical frequencies tractable in the presence of negated relations by the inverse Möbius transform. Evaluation of our method on four benchmark datasets shows that maximum pseudo-likelihood provides fast and accurate estimates at different sample sizes.

1 Introduction

Many applications store data in relational form. Relational data introduces the machine learning problem of *class-level frequency estimation*: building a model that can answer statistical queries about classes of individuals in the database [2]. Examples of such queries include:

- What fraction of birds fly?—referring to the class of birds.
- What fraction of the grades awarded to highly intelligent students are As?—referring to the class of (student, course-grade) pairs.
- In a social network, what is the percentage of friendship pairs where both are women?—referring to the class of friendship links.

As the examples illustrate, class-level probabilities concern the frequency, proportion, or rate of generic events and conditions, rather than the attributes and links of individual entities. Estimates of class-level frequency have several applications:

Statistical first-order Patterns. AI research into combining first-order logic and probability investigated in depth the representation of statistical patterns in relational structures [3, 4]. Often such patterns can be expressed as *generic statements* about the average member of a class, like "intelligent students tend to take difficult courses".

Policy making and strategic planning. A university administrator may wish to know which program characteristics attract high-ranking students in general, rather than predict the rank of a specific student in a specific program. Maier *et al.* [5] describe several applications of causal-relational knowledge for decision making.

Query optimization. A statistical model predicts a probability for given table join conditions that can be used to estimate the cardinality of a database query result [2]. Cardinality estimation is used to minimize the size of intermediate join tables [6].

Semantics. We build a Bayes net model for relational statistics, using the Parametrized Bayes nets (PBNs) of Poole [7]. The nodes in a PBN are constructed with functors and first-order variables (e.g., $gender(X)$ may be a node). We extend the PBN semantics to handle the ungrounded cases. The original PBN semantics is a grounding semantics where the first-order Bayes net is instantiated with all possible groundings to obtain a directed graph, whose nodes are functors with constants (e.g., $gender(sam)$). The ground graph can be used to answer queries *about individuals*, such as "if user *sam* has 3 friends, female *rozita*, males *ali* and *victor*, what is the probability that *sam* is a woman"? However, as Getoor pointed out [8], the ground graph is not appropriate for answering class-level queries, which are about generic rates and percentages, not about any particular individuals.

We support class-level queries via a new grounding-free interpretation for Parametrized Bayes nets. The semantics is based on Halpern's classic random selection semantics for probabilistic first-order logic [3, 4]. Halpern's semantics views statements with first-order variables as expressing statistical information about classes of individuals. For instance, the claim "60% is the percentage of friendship pairs where both friends are women" could be expressed by the formula

$$P(\mathit{Gender}(X) = \mathit{female}, \mathit{Gender}(Y) = \mathit{female} | \mathit{Friend}(X, Y)) = \mathit{60}\%.$$

Learning. A standard Bayes net parameter learning method is maximum likelihood estimation, but this method is difficult to apply for Bayes nets that represent relational data because the cyclic data dependencies in relations violate the requirements of a traditional likelihood measure. We circumvent these limitations by using a relational pseudo-likelihood measure for Bayes nets [1] that is well defined even in the presence of cyclic dependencies. In addition to this robustness, the relational pseudo-likelihood matches the random selection semantics because it is also based on the concept of random instantiations. The estimator that maximizes the parameters of this pseudo-likelihood function (MPLE) has a closed-form solution: the parameters are the empirical frequencies, as with classical i.i.d. maximum likelihood estimation. Computing the empirical frequencies for negated relationships is difficult, however, because enumerating the complement of a relationship table is computationally infeasible. We show that the Möbius transform [9] makes MPLE tractable for negated relationships.

Results. We evaluate MPLE on four benchmark real-world datasets. On complete-population samples MPLE achieves near perfect accuracy in parameter estimates, and excellent performance on Bayes net queries. The accuracy of MPLE parameter values is high even on medium-size samples.

Contributions. Our contributions for frequency modelling in relational data are the following:

1. A new grounding-free semantics for graphical first-order models that supports class-level queries. While we focus on Bayes net models, the semantics applies generally to statistical-relational models that are based on first-order logic.
2. A novel application of the Möbius transform that provides a tractable algorithm for estimating class-level probabilities from an input database. While we focus on Bayes net parameter learning, this algorithm is a general procedure for computing relational counts that involve negated links. It has application in Probabilistic Relational Models [10, Sec.5.8.4.2], multi-relational data mining, and inductive logic programming models with clauses that contain negated relationships.

Paper Organization. We begin with background in probability logic (Section 2). Section 3 presents relational Bayes net models and the random selection semantics for Bayes nets. Section 4 describes parameter learning and Section 5 presents the Möbius transform for computing relational statistics. Simulation results are presented in Section 6, showing the runtime cost of estimating parameters, together with evaluations of their accuracy. We summarize related work in Section 7.

2 Background: Predicate and Probability Logic

Our work synthesizes concepts from logic, database theory, probability theory, and Bayes nets. We review only the concepts that are the focus of this paper, with an emphasis on semantics. Our results can be extended to richer frameworks.

2.1 Predicate Logic

We adopt Chiang and Poole's version of predicate logic for statistical-relational modelling [11].

Syntax. Individuals are represented by **constants**, which are written in lower case (e.g., *cs100*). A **type** is associated with each individual, for instance *bob* is a *person*. The **population** associated with type τ, a set of individuals, is denoted by $\mathcal{P}(\tau)$. We assume that the populations are specified as part of the type description. A **population variable** is capitalized and associated with a type. For example, the population variable *Person* is associated with $\tau = \mathit{person}$.

A **functor** represents a mapping $f : \Omega_f \rightarrow V_f$ where f is the name of the functor, $\Omega_f \equiv \mathcal{P}(\tau_1), \ldots, \mathcal{P}(\tau_a)$ is the *domain of the relation*, and $\tau_1, \ldots, \tau_a$ are the *argument types* of the functor. The output type or *range* of the functor is denoted by V_f. In this paper we consider only functors with a finite range, disjoint from all populations. If $V_f = \{T, F\}$, the functor f is a (Boolean) **predicate**; other functors are called **attributes**. A predicate with more than one argument is called a **relationship**.

An **atom** is an expression of the form $f(\sigma_1, \ldots, \sigma_a)$, where each σ_i is either a constant or a population variable [11]. The types of $\sigma_1, \ldots, \sigma_a$ must match the argument types of f. If all the $\sigma_1, \ldots, \sigma_a$ are constants, then $f(\sigma_1, \ldots, \sigma_a)$ is a **ground atom**. A **literal** specifies the value of an atom, generically $f(\sigma_1, \ldots, \sigma_a) = v$, where $v \in V_f$. A literal is the basic unit of information. Literals can be combined to generate *formulas.* In this paper we consider only formulas that are **conjunctions** of literals, denoted by the Prolog-style comma notation $f(\sigma_1, \ldots, \sigma_a) = v, \ldots, f'(\sigma'_1, \ldots, \sigma'_{a'}) = v'$.

Let $\{X_1, \ldots, X_k\}$ be a set of population variables with types $\tau_1, \ldots, \tau_k$. A **grounding** γ is a set $\gamma = \{X_1 \backslash x_1, \ldots, X_k \backslash x_k\}$ where each x_i is a constant of the same type as X_i. A grounding γ for a formula ϕ is a grounding for the variables that appear in the formula, denoted by $\phi\gamma$.

In classic AI papers connecting logic and probability, Halpern and Bacchus extended predicate logic to incorporate statements about probabilities [3, 4]. Formalizing probability within logic provides a formal semantics for probability assertions and allows principles for probabilistic reasoning to be defined as logical axioms.

Examples. Consider a social network model with a single type, $\tau = \mathit{people}$, associated with two population variables X and Y. The population $\mathcal{P}(\mathit{people})$ comprises a set of individual people, for instance $\{\mathit{anna}, \mathit{bob}\}$. The functor

$$\mathit{gender} : \mathcal{P}(\mathit{people}) \rightarrow \{M, W\}$$

takes as input a person and returns either M for “man” or W for “woman”. The predicate

$$\mathit{Friend} : \mathcal{P}(\mathit{people}) \times \mathcal{P}(\mathit{people}) \rightarrow \{T, F\}$$

indicates whether two people are friends.

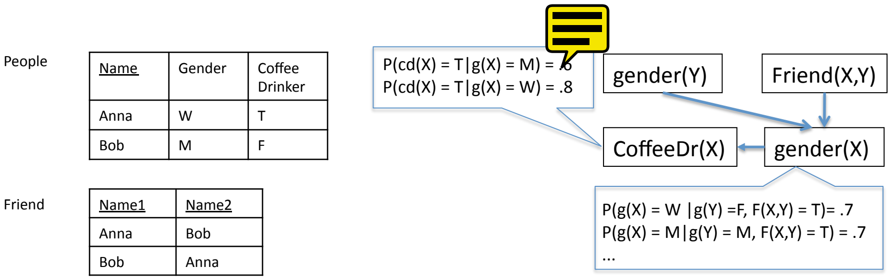
Nonground atoms in this language include $\mathit{gender}(X)$, $\mathit{gender}(Y)$,and $\mathit{Friend}(X, Y)$. Ground atoms include $\mathit{gender}(\mathit{anna})$ and $\mathit{Friend}(\mathit{anna}, \mathit{bob})$. A conjunction of nonground literals is $\mathit{gender}(X) = W$, $\mathit{gender}(Y) = M$, $\mathit{Friend}(X, Y) = T$.

Applying the grounding $\{X\backslash anna, Y\backslash bob\}$ produces the conjunction of ground literals $\mathit{gender}(\mathit{anna}) = W$, $\mathit{gender}(\mathit{bob}) = M$,$\mathit{Friend}(\mathit{anna}, \mathit{bob}) = T$.

Semantics. In logic, semantics defines the rules for determining the truth of a sentence with respect to a particular relational structure [13, Ch.7.4.2]. A **relational structure** $\mathcal{S}_f$ for functor f is a set of tuples $\{\langle x_1, \ldots, x_a, v\rangle\}$, where the tuple $\langle x_1, \ldots, x_a\rangle$ is in the domain of f, and the value v is in the range of f. Each tuple in the domain of f appears exactly once in $\mathcal{S}_f$. A relational structure $\mathcal{S}$ for a functor language contains one relational structure for each functor. If the structure $\mathcal{S}$ assigns the same value to a ground atom as a literal l, the literal is true in $\mathcal{S}$, written as $\mathcal{S} \models l$. Thus

$$\mathcal{S} \models f(x_1, \ldots, x_a) = v \text{ iff } \langle x_1, \ldots, x_a, v\rangle \in \mathcal{S}_f.$$

A conjunction is true if all its conjuncts are true. A relational structure can be represented as one or more tables (Figure 1). All ground literals listed above are true in the structure of Figure 1. False literals include $\mathit{Coffee\_Drinker}(\mathit{bob}) = T$ and $\mathit{Friend}(\mathit{anna}, \mathit{anna}) = T$.



People

Name	Gender	Coffee Drinker
Anna	W	T
Bob	M	F

Friend

Name1	Name2
Anna	Bob
Bob	Anna

P(cd(X) = T|g(X) = M) = .6
P(cd(X) = T|g(X) = W) = .8
gender(Y)
Friend(X,Y)
CoffeeDr(X)
gender(X)
P(g(X) = W |g(Y) =F, F(X,Y) = T)= .7
P(g(X) = M|g(Y) = M, F(X,Y) = T) = .7
...

Fig. 1. Left: A relational structure represented as tables. By convention, a pair not listed in the *Friend* table is not a pair of friends. Right: A Bayes Net for this database. $\mathit{Friend}(X, Y)$ is a relationship node, while the other three are attribute nodes. The conditional probabilities are not related to the database.

A relational structure does not determine a truth value for a nonground literal such as $\mathit{gender}(X) = M$, since a gender is not determined for a generic person X. We follow database theory and view a nonground formula as a *logical query* [14]. Intuitively, the formula specifies conditions on the population variables that appear in it, and the result set lists all individuals that satisfy these conditions in a given relational structure. Formally, we denote the formula **result set** for a structure S as $\mathbb{R}_{\mathcal{S}}(\phi)$, defined by

$$\mathbb{R}_{\mathcal{S}}(\phi) \equiv \{\langle x_1, \ldots, x_k\rangle : \mathcal{S} \models \phi\{X_1\backslash x_1, \ldots, X_k\backslash x_k\}\rangle$$

If we extend our predicate logic with disjunctions, and restrict negations in a natural way, we obtain a logical query language known as the *safe domain* *relational calculus* [14]. A fundamental result of database theory states that the result sets definable in the safe domain relational calculus are exactly the same as those definable in standard relational algebra, using table joins.

The **cardinality** of a query formula in a structure S is simply the size of its resultset: $\#_{\mathcal{S}}(\phi) \equiv |\mathbb{R}_{\mathcal{S}}(\phi)|$. The *cardinality estimation problem* is to estimate the size of a result set, without actually retrieving the tuples in the set. Cardinality estimation is key for optimizing query processing, and has therefore been studied extensively by database researchers [6]. Getoor, Taskar, and Koller [2] observed that this estimation problem is equivalent to the *query probability estimation problem*, defined by dividing the cardinality of a query's resultset by its maximum size:

$$P_{\mathcal{S}}(\phi) \equiv \frac{\#_{\mathcal{S}}(\phi)}{|\mathcal{P}(\tau_1)| \times \cdots \times |\mathcal{P}(\tau_k)|} \tag{1}$$

where $\tau_1, \ldots, \tau_k$ are the types of the k population variables that appear in ϕ. For example, in the structure of Figure 1, we have

$$P_{\mathcal{S}}(\mathit{Friend}(X,Y) = T, \mathit{Gender}(X) = M, \mathit{Gender}(Y) = F) = \mathit{1/4}$$

since the gender conditions are met in 1 out of 4 possible instantiations of the X, Y population variables. The advantage of the probability formulation is that it casts cardinality estimation as a statistical density estimation problem. Getoor [8] developed a Bayes net-type model for query probability estimation.

2.2 Probability Logic

In this section we extend the predicate logic presented above with concepts from probability logic as developed by Bacchus [4] and Halpern [3]. A **probability formula** is a conjunction of the form

$$P(\phi_1) = p_1, \ldots, P(\phi_n) = p_n$$

where each p_i is a term that denotes a real number in the interval [0,1] and each ϕ_i is a conjunctive formula in predicate logic as defined above. We refer to predicate logic extended with probability formulas as *probability logic.*

Halpern presents a different semantics for probability formulas depending on whether they contain only ground literals, only nonground literals, or both. An example of a nonground probability formula is $P(\mathit{Flies}(B) = T) = \mathit{90}\%$, and an example of a ground probability formula is $P(\mathit{Flies}(\mathit{tweety}) = \mathit{90}\%$. The semantics for probability formulas with ground literals is based on a probability distribution over possible relational structures. This *possible world semantics* for ground literals is well-known in the statistical-relational learning community [15, 12],[13, Ch.14.6.1]. Less well-known is Halpern's **random selection semantics** semantics for nonground literals. In this paper we use probability formulas with nonground literals to represent class-level probabilities, so we describe their semantics in some detail.

A relational structure is augmented with a distribution P_τ over each population $\mathcal{P}(\tau)$. We may then view a population variable as *a random selection* from its population. These random selections are independent, so for a fixed set of population variables, the population distributions induce a distribution over the groundings of the variables. We assume that all individuals are treated equally, and hence assume a uniform distribution over each population.[1] This leads to the following definitions:

1. $P_{\mathcal{S}}(X = x) \equiv P_\tau(x) = \frac{1}{|\mathcal{P}(\tau)|}$.
2. $P_{\mathcal{S}}(X_1 = x_1, \ldots, X_k = x_k) \equiv P_{\tau_1}(x_1) \times \cdots \times P_{\tau_k}(x_k) = \frac{1}{|\mathcal{P}(\tau_1)| \times \cdots \times |\mathcal{P}(\tau_k)|}$

where τ_i is the type of variable X_i and constant x_i. The random selection semantics states that the probability of a nonground formula is the probability of the set of groundings that satisfy the formula.

$$\mathcal{S} \models P(\phi) = p \text{ iff } p = \sum_{\langle x_1, \ldots, x_k \rangle \in \mathbb{R}_{\mathcal{S}}(\phi)} P_{\mathcal{S}}(x_1, \ldots, x_k) \tag{2}$$

Under our assumption of uniform population distributions, the random selection semantics probability for a formula is identical to its query probability $P_{\mathcal{S}}(\phi)$ (Equation 1).

Interpretation. As the random selection semantics is a key concept in this paper, we discuss several interpretations.

Atoms as Random Variables. Population variables are random variables selecting population members. Nonground atoms represent functions applied to the result of the selection. *Since a function of random variables is also a random variable,* we can view nonground atoms as random variables. Figure 2 illustrates this view.

Random Individuals. Under random selection semantics, probability formulas can be interpreted as describing the properties of *random individuals.* For example, the formula $P(\mathit{Flies}(B)) = \mathit{90}\%$ can be interpreted as "the probability that a randomly selected bird flies is 90%". In many domains, an intuitive interpretation is in terms of typical individuals; for instance, we may read the formula as "the probability that a typical bird flies is 90%".

Class-level Probabilities. Random selection formulas can also be read as describing the distribution of properties in *classes of individuals.* Thus $P(\mathit{Flies}(B)) = \mathit{90}\%$ can be read as "the percentage of birds that fly is 90%". We therefore refer to probabilities defined by random selection as **class-level probabilities**. In the next section we apply the random selection semantics to show how Bayes nets can be interpreted as specifying class-level probabilities.

3 Bayes Nets for Relational Data

A **Bayes Net (BN)** is a directed acyclic graph (DAG), whose nodes comprise a set of random variables and conditional probability parameters. The joint

[1] Halpern allows nonuniform distributions.

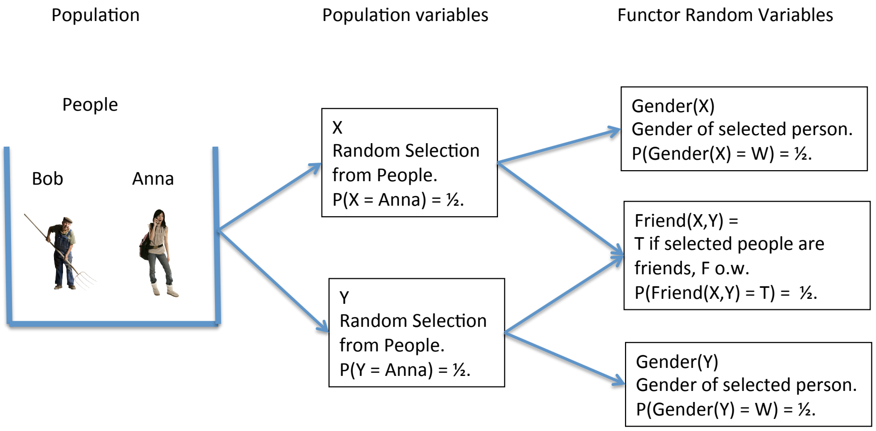


Fig. 2. Viewing population variables as random selections from the associated population, functor nodes are functions of random variables, hence themselves random variables. Probabilities are computed from the relations in Fig. 1.

probability to each assignment of values to the nodes in the BN is specified by the standard product formula, namely the product of the conditional probabilities $P(child|parent\_values)$. A **Parametrized Bayes Net** (PBN) is a Bayes net whose nodes are atoms [7]. We also refer to a PBN as a **Functor Bayes Net** and to its atoms as **functor nodes**. In the rest of this paper, we will only consider Parametrized Bayes Nets and refer to them as simply Bayes nets. This briefer notation avoids confusion with the more common statistical sense of "parametrized", meaning that values have been assigned to the parameters of the Bayes net. In this paper we consider Bayes nets with nonground atoms only. The right side of Figure 1 shows an illustrative Bayes net.

Before learning the parameters of a Bayes net, we must establish its DAG. A **Bayes net structure** is a DAG with no probabilities associated with its nodes. Bayes net structures can be either specified *a priori* by the analyst or learned using one of several algorithms. This paper is only concerned with learning Bayes net parameters and using them in inference; in our evaluations (Section 6), we use previously-published algorithms to first learn the Bayes net structure.

Class-Level Semantics for Bayes Nets. Random selection semantics can be applied to view Bayes Nets as representing class-level probabilities, without reference to a ground model. Via the standard product formula, a Bayes net B entails a probability value P_B for each joint assignment of values to functor nodes. These are probability formulas with nonground literals. For example, the Bayes net of Figure 1 entails a set of joint probabilities of the form

$$P_B(G(Y) = v_1, F(X,Y) = v_2, G(X) = v_3, CDr(X) = v_4),$$

where we have used the obvious abbreviations for functors, and each v_i is a constant from the appropriate domain. To illustrate, with uniform distributions on $G(Y)$ and $F(X, Y)$, the Bayes net of Figure 1 entails the probability

$$P_B(G(X) = W, G(Y) = M, F(X, Y) = T, CD(X) = T) = 0.06. \tag{3}$$

The random selection provides two related interpretations of these joint probabilities. (1) As shown in Figure 2, if we view each population variable as a random selection from its domain, then each functor node is a function of one or more random variables, and hence itself a random variable. The Bayes net model therefore represents a joint distribution over random variables as usual. The extension from the nonrelational setting is simply that a random variable in the model is now a complex object with internal syntactic structure, rather than a simple "flat" object. The random selection semantics provides a meaning for these structured random variables. For example, using the obvious abbreviations for the BN of Figure 1, the joint probability (3) can be read as "if we randomly select two users X and Y, the chance that one is a woman, the other is a man, they are friends, and the woman drinks coffee is 6%".

(2) Each joint probability specified by the Bayes net is a conjunction in probability logic. The Bayes net therefore specifies a set of nonground probability formulas. In the terminology of Bacchus, Grove, Koller, and Halpern, such a set is a *statistical knowledge base* [16]. In this view, a Functor Bayes net is a compact graphical representation of a statistical knowledge base, much as a small set of axioms can provide a compact representation of a logical knowledge base.

4 Parameter Learning For Class-Level Probabilities

The data for relational learning constitute a relational substructure drawn from a complete structure. When the substructure represents observed data, we refer to it as a **database** and use the notation $\mathcal{D}$ instead of $\mathcal{S}$. We make the standard assumption that the database is complete [17, 18]: For each individual (node) observed, the database lists its attributes, and for each pair of observed individuals (nodes), the database specifies which links hold between them. Note that completeness is a statement about the information about individuals in the database. A complete database may contain only a subset of the individuals in the complete structure.

A fundamental question for statistical model selection is how to measure the fit of a Bayes net model to a database. That is, we must specify a likelihood function $P_B(\mathcal{D})$ for a given database. The random selection semantics can be used to define a tractable *pseudo-likelihood* [1]: The pseudo log-likelihood is the expected log-probability assigned by the Bayes net to the links and attributes for a *random grounding* of the population variables. This is defined by the following procedure.

1. Randomly select a grounding for *all* 1st-order variables that occur in the Bayes Net. The result is a ground graph with as many nodes as the original Bayes net.

2. Look up the value assigned to each ground node in the database. Compute the log-likelihood of this joint assignment using the usual product formula; this defines a log-likelihood for the random instantiation.
3. The expected value of this log-likelihood is the *pseudo log-likelihood* of the database given the Bayes net.

X	Y	F(X,Y)	G(X)	CD(X)	G(Y)	BN probability	BN log-p
Anna	Bob	T	F	T	M	$0.5^2 \cdot 0.3 \cdot 0.8 = 0.06$	-2.81
Bob	Anna	T	M	F	F	$0.5^2 \cdot 0.3 \cdot 0.6 = 0.24$	-3.10
Anna	Anna	F	F	T	F	$0.5^2 \cdot 0.5 \cdot 0.8 = 0.26$	-2.30
Bob	Bob	F	M	F	M	$0.5^2 \cdot 0.5 \cdot 0.6 = 0.11$	-2.59

Table 1. An example computation of the pseudo-likelihood for the database and the Bayes net of Figure 1. (Unspecified BN parameters are chosen as uniform solely for computational convenience.) The pseudo log-likelihood is -2.7, the average of the log-probabilities (rightmost column).

Table 1 shows an example computation of the pseudo likelihood. The pseudo likelihood matches the random selection semantics that we have proposed for class-level Bayes nets. It has several attractive theoretical properties [1]. (1) It is closely related to other relational pseudo-likelihood measures proposed for graphical models (e.g., Markov random fields). (2) It is invariant under equivalence transformations of the logical predicates (database normalization). (3) For a fixed database $\mathcal{D}$ and Bayes net structure, *the parameter values that maximize the pseudo-likelihood are the conditional empirical frequencies* defined by the database distribution $P_{\mathcal{D}}$ [1, Prop.3.1]. This result is exactly analogous to maximum likelihood estimation for i.i.d. data. In the remainder of the paper we evaluate database frequencies as parameter estimates.

5 Computing Relational Statistics

In a Bayes net, a **family** is a child node together with its parents [13, Ch.14.2]. We consider the problem of computing the database joint probability of a non-ground formula that describes a family state, $P_{\mathcal{D}}(\mathit{child\_value}, \mathit{parent\_values})$. Joint probabilities are easily converted to conditional probability parameters in the Bayes net. The computation depends on whether the family formula contains negated relationships. An example of only positive relationships is $P_{\mathcal{D}}(\mathit{gender}(X) = M, \mathit{Friend}(X,Y) = T)$, the frequency of friendship pairs (x, y) where x is male. The required count $\#_{\mathcal{D}}(\mathit{gender}(X) = M, \mathit{Friend}(X,Y) = T)$ can be computed by regular joins or optimized virtual joins [19].

5.1 Statistics With One Negated Relationship

A Functor Bayes net represents *uncertainty about relationships* with relationship nodes such as $\mathit{Friend}(X, Y)$. Learning with link uncertainty requires computing sufficient statistics that involve the *absence* of relationships. A naive approach would explicitly construct new data tables that enumerate tuples of objects that are *not* related. However, the number of unrelated tupes is too large to make this scalable (think about the number of user pairs who are *not* friends on Facebook). Getoor et al. observed that we can use a "1-minus trick" for the special case of a query formula with only a single negated relationship [10], however this method does not extend to more than one negated relationship.

5.2 Statistics With Multiple Negated Relationships: The Fast Möbius Transform

The general case of multiple negative relationships can be computed using the **inverse Möbius transform** (IMT). This algorithm is optimal for computing a complete set of joint probabilities with any number of positive and negative relationship from the probability values for formulas with only positive relationships [9]. We adapt the original lattice-structure formulation of IMT for **joint probability tables** (JP-tables), CP-tables where the conditional probabilities are replaced by joint probabilities.

Consider a set $R_1, \ldots, R_m$ of Boolean relationship atoms, and a set $f_1, \ldots, f_k$ of attribute atoms. The goal is to compute joint probability estimates $P(R_1 = b_1, \ldots, R_m = b_m, f_1 = v_1, \ldots, f_k = v_k)$ for all possible Boolean values $b_i \in \{T, F\}$, and all possible values v_i in the range of f_i.

The algorithm first estimates all marginal probabilities that involve only positive values; these are called the **Möbius parameters** of P [20, p.239]. For simplicity, consider a fixed conjunction of attribute literals ϕ. Then we compute 2^m estimates such as $P(R_1 = T, \phi)$; $P(R_1 = T, R_2 = T, \phi)$; $P(R_1 = \mathit{true}, R_3 = T, R_m = T, \phi)$; etc. To represent a Möbius parameter, we allow relationship nodes to take on the value $*$ for "unspecified". For instance, suppose that the atoms are $\mathit{Int}(S)$, $\mathit{Reg}(S, C)$, $\mathit{RA}(S, P)$. Then the Möbius parameter $P(\mathit{Int}(S) = \mathit{1})$ is stored in the JP-table row where $\mathit{Int}(S) = \mathit{1}, \mathit{Registered}(S, C) = *$, and $\mathit{RA}(S, P) = *$. The *Möbius extension theorem* shows that the Möbius parameters entail a unique set of values for all joint probabilities [20, p.239]. To compute these values, the IMT uses a local update operation corresponding to

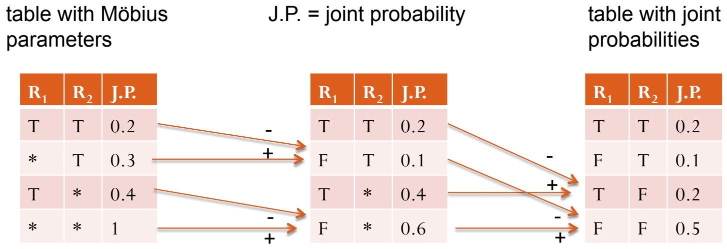
$$P(\phi, \mathbf{R}, R = F) := P(\phi, \mathbf{R}) - P(\phi, \mathbf{R}, R = T). \tag{4}$$

where $\mathbf{R}$ is a conjunction of relationship literals that assign Boolean values to relationship atoms.

Eq. 4 generalizes the 1-minus trick, computing a probability with $k + 1$ negated relationships from two probabilities that each have only k negated relationships. The IMT initializes the JP-table with the Möbius parameters that do not contain negated relationships, that is, all relationship nodes have the value

Algorithm 1 The inverse Möbius transform for parameter estimation in a Parametrized Bayes Net.

```
Input: database $\mathcal{D}$; a set of functor nodes divided into attribute nodes $f_1, \ldots, f_j$ and relationship nodes $R_1, \ldots, R_m$.
Output: Joint Probability specifying the data frequencies for each joint assignment to the input functor nodes.
for all attribute value assignments $f_1 := v_1, \ldots, f_j := v_j$ do
    initialize the JP-table with the Möbius parameters: set all relationship nodes to either $T$ or $*$; find joint frequencies with data queries.
    for $i = 1$ to $m$ do
        Change all occurrences of $R_i = *$ to $R_i = F$.
        Update the joint frequencies using (4).
    end for
end for
```



R_1	R_2	J.P.
T	T	0.2
*	T	0.3
T	*	0.4
*	*	1

R_1	R_2	J.P.
T	T	0.2
F	T	0.1
T	*	0.4
F	*	0.6

R_1	R_2	J.P.
T	T	0.2
F	T	0.1
T	F	0.2
F	F	0.5

Fig. 3. The inverse Möbius transform with $m = 2$ relationship nodes. For simplicity we omit attribute conditions.

T or $*$. It then goes through the relationship nodes $R_1, \ldots, R_m$ in order, at stage i replacing all occurrences of $R_i = *$ with $R_i = F$, and applying the local update equation for the probability value for the modified row. At termination, all $*$ values have been replaced by F and the JP-table specifies all joint frequencies. Algorithm 1 gives pseudocode and Figure 3 presents a small example.

5.3 Complexity Analysis.

Kennes and Smets [9] analyze the complexity of the IMT. We summarize the main points. (1) The IMT *accesses data only about existing links*, never about nonexisting links. (2) The number of additions performed is $m2^{m-1}$, where m is the number of relationship nodes. This is the lower bound [9, Cor.1]. (3) For general m, computing sufficient statistics in a relational structure is #P-complete [21, Prop.12.4]. In practice, the number m is small, 4 or less.

6 Evaluation

All experiments were done on a QUAD CPU Q6700 with a 2.66GHz CPU and 8GB of RAM. The datasets and code are available on the Web [22].

6.1 Datasets

We used four benchmark real-world databases, with the modifications by Khosravi et al. [23]. See that article for details.

Mondial Database. A geography database, featuring a self-relationship, *Borders*, that indicates which countries border each other.

Hepatitis Database. A modified version of the PKDD'02 Discovery Challenge database.

Financial A dataset from the PKDD 1999 cup.

MovieLens. A dataset from the UC Irvine machine learning repository. We also conducted experiments with synthetic graph and datasets. The results are similar to those on real-life datasets. We omit details because of space constraints.

To obtain a Bayes net structure for each dataset, we applied the learn-and-join algorithm [23] to each database. This is the state-of-the-art structure learning algorithm for PBNs; for an objective function, it uses the pseudo-likelihood described in Section 4. The algorithm was implemented using version 4.3.9-0 of CMU's Tetrad package [24].

6.2 Learning Times

Table 2 shows the runtimes for computing parameter values. The Complement method uses SQL queries that explicitly construct tables for the complement of relationships (tables representing tuples of *unrelated* entities), while the IMT method uses the inverse Möbius transform to compute the conditional probabilities. The IMT is faster by factors of 15–237.

Table 2. Learning times (sec) for algorithms that construct explicit complement tables or use the inverse Möbius transform.

Database	Parameters	#tuples	Complement	IMT	Ratio
Mondial	1618	814	157	**7**	22
Hepatitis	1987	12,447	18,246	**77**	237
Financial	10926	17,912	228,114	**14,821**	15
MovieLens	326	82,623	2,070	**50**	41

6.3 Inference

Parameter learning for Bayes nets is evaluated based on query performance of the resulting Bayes nets [25] because answering probabilistic queries is the basic inference task for Bayes nets. Correct parameter values will support correct query results, so long as the given Bayes net structure matches the true distribution (i.e., is an I-map). We evaluate our algorithm by formulating random queries and comparing the results returned by our algorithm to the ground truth, the actual database frequencies.

For each dataset, 100 test queries were randomly generated according to the following procedure. First, choose a target node V, and enumerate all values a in the range of V. For each value a, choose the number k of conditioning variables, uniformly distributed from 1 to 3. Select k variables $V_1, \ldots, V_k$ and corresponding values $v_1, \ldots, v_k$, where each v_i is chosen from the range of V_i according to the uniform distribution. The query to be answered is then $P(V = a|V_1 = v_1, \ldots, V_k = v_k)$.

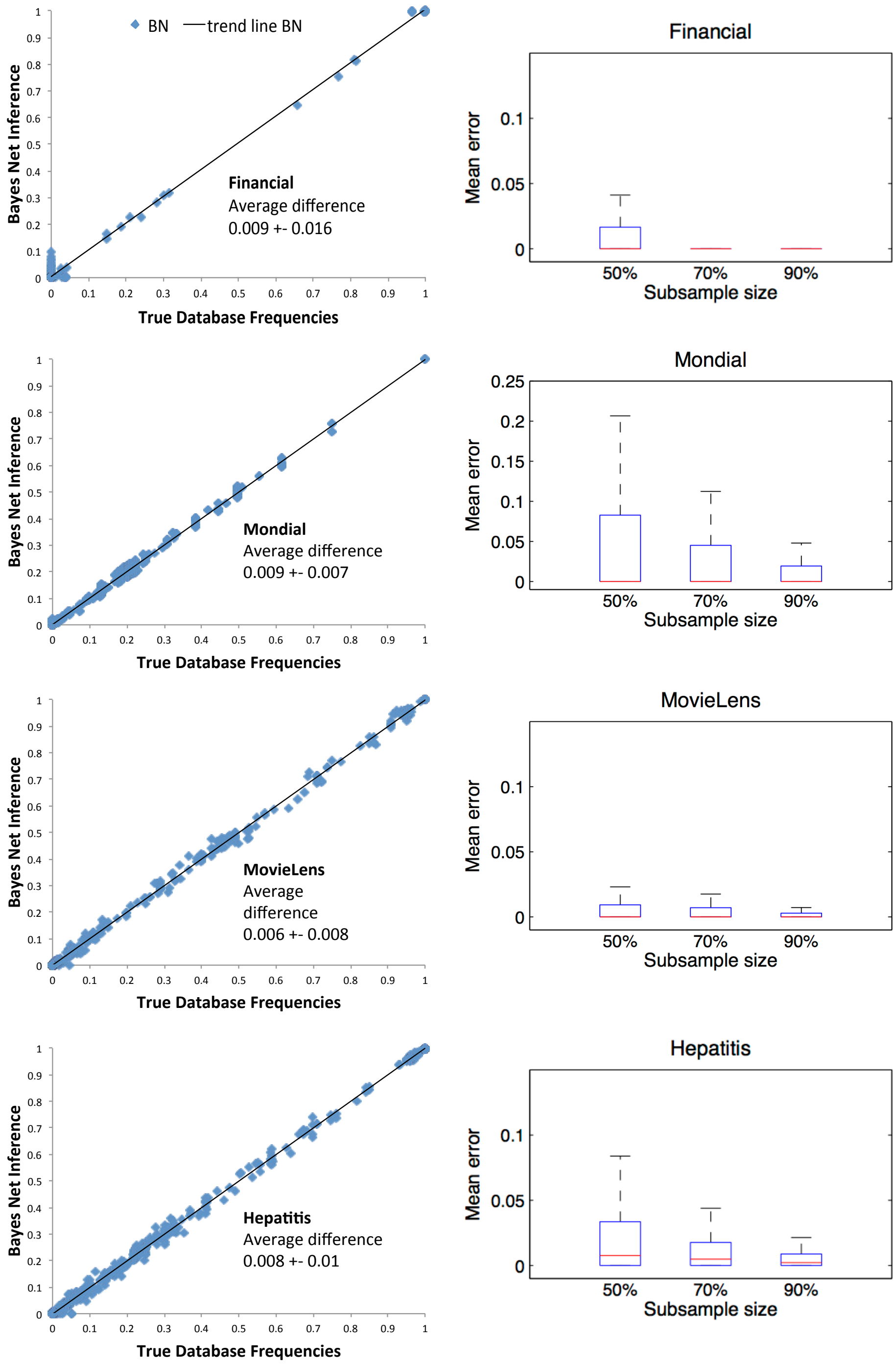
As done by Getoor et al. [2], we evaluate queries after learning parameter values on the entire database. Thus the Bayes Net is viewed as a statistical summary of the data rather than as generalizing from a sample. Inference is carried out using the Approximate Updater in CMU's Tetrad program. Figure 4 shows the query performance for each database. A point (x, y) on a curve indicates that there is a query such that the true probability value in the database is x and the probability value estimated by the model is y. The Bayes net inference is close to the ideal identity line, with an average error of less than 1%.

Comparison With Markov Logic Networks. To benchmark our results, we compare the average error of PBN inferences with frequency estimate from Markov Logic Network (MLN)s. MLNs are a good comparison point because (1) they are currently one of the most active areas of SRL research; (2) the Alchemy system provides an open-source, state-of-the-art implementation for learning and inference [26]; (3) they do not require the specification of further components (e.g., a combining rule or aggregation function); and (4) their undirected model can accommodate recursive relationships (cyclic dependencies).

Like most statistical-relational models, MLNs were designed for instance-level probabilities rather than class-level queries. We therefore need an extra step to convert instance-level probabilities into class-level probabilities. To derive class-level probability estimates from instance-level inference, the following method due to Halpern [3, fn.3] can be used: Introduce a new individual constant for each 1st-order variable in the relational model (e.g., random-student, random-course, and random-prof). Applying instance probability inference to these new individuals provides a query answer that can be interpreted as a class-level probability. To illustrate, if the only thing we know about Tweety is that Tweety is a bird, then the probability that Tweety flies should be the frequency of flyers in the bird population [27]. That is, the class-level probability $P(\mathit{Flies}(B) = T)$ can be obtained from the instance-level probability $P(\mathit{Flies}(\mathit{tweety}) = T)$ where *tweety* is a new constant not featured in the data used for learning. In the experiments below, we apply MLN inference to new constants as described to obtain class-level probability estimates. We compare the following learning algorithms [28]:

BN: Bayes net parametrized with maximum pseudo likelihood estimates.
LHL: A structure learning algorithm that produces a parametrized MLN, with Alchemy's MC-SAT for inference.
LSM: Another structure learning algorithm that produces a parametrized MLN,

Fig. 4. *Left*: Query Performance: Estimated vs. true probability, with average error and standard deviation. Number of queries/average inference time per query: Mondial, 506/0.08sec; MovieLens, 546/0.05sec; Hepatitis, 489/0.1sec; Financial, 140/0.02sec. *Right*: Error (absolute difference) in estimated conditional probability parameters, averaged over 10 random subdatabases and all BN parameters.



with Alchemy's MC-SAT for inference. In experiments by Kok and Domingos, LSM outperformed previous MLN learners.

As shown in Table 3, the Bayes net models provide an order of magnitude more accurate class-level estimates than the MLN models.

	Average error		
Dataset	BN	MLN(LSM)	MLN(LHL)
Mondial	**0.9**%	8.6%	10.5%
Hepatitis	**0.8**%	11.2%	13.2%
Financial	**0.9**%	9.1%	NT
Movielens	**0.6**%	14.2%	NT

Table 3. Class-level query performance of Bayes nets vs. two algorithms for learning Markov Logic Networks, as indicated by average absolute error between the predicted frequency and the true database frequencies. NT denotes non-termination within the system resources.

6.4 Conditional Probabilities

In the previous inference results, the correct query answers are defined by the entire population, represented by the benchmark database, and the model is also trained on the entire database. To study parameter estimation at different sample sizes, we trained the model on N% of the data for varying N while continuing to define the correct value in terms of the entire population. Conceptually, we treated each benchmark database as specifying an entire population, and considered the problem of estimating the complete-population frequencies from partial-population data. The N% parameter is uniform across tables and databases. We employed standard subgraph subsampling [29, 23], which selects entities from each entity table uniformly at random and restricts the relationship tuples in each subdatabase to those that involve only the selected entities. Subgraph sampling provides complete link data and matches the random selection semantics. It is applicable when the observations include positive and negative link information (e.g., not listing two countries as neighbors implies that they are not neighbors). The subgraph method satisfies an ergodic law of large numbers: as the subsample size increases, the subsample relational frequencies approach the population relational frequencies.

The right side of Figure 4 plots the difference between the true conditional probabilities and the MPLE estimates. With increasing sample size, **MPLE** estimates approach the true value in all cases. Even for the smaller sample sizes, the median error is close to 0, confirming that most estimates are very close to correct. As the box plots show, the 3rd quartile of estimation error is bound within 10% on Mondial, the worst case, and within less than 5% on the other datasets.

7 Related Work and Discussion

Class-level vs. Instance-level Probabilities. Most previous work on statistical-relational learning has been concerned with instance-level probabilities for predicting the attributes and relationships of *individuals* [30, 13]. Examples of instance-level queries include the following.

- Given that Tweety is a bird, what is the probability that Tweety flies?
- Given that Sam and Hilary are friends, and given the genders of all their other friends, what is the probability that Sam and Hilary are both women?
- What is the probability that Jack is highly intelligent given his grades?

In probability logic, class-level probabilities are represented by formulas with nonground literals (e.g., $P(\mathit{Flies}(B)) = p$), and instance-level probabilities are represented by formulas with ground literals (e.g., $P(\mathit{Flies}(\mathit{tweety})) = p$). In graphical models, instance level probabilities are defined by a *template semantics* [30]: The class-level model is instantiated by grounding all functor nodes with every appropriate constant. Instance-level probabilites can be computed by applying inference to the ground graph.

A key difference between class-level and instance-level queries is that instance-level queries can involve an unbounded number of random variables. For instance, if *maite* has 100 Facebook friends $x_1, \ldots, x_{100}$, an instance-level query to predict the gender of *maite* given the genders of her friends would involve 100 ground literals of the form

$$P(\mathit{gender}(\mathit{maite}) | \mathit{gender}(x_1), \mathit{Friend}(\mathit{maite}, x_1), \ldots, \mathit{gender}(x_{100}), \mathit{Friend}(\mathit{maite}, x_{100}).$$

In contrast, class-level queries are restricted to the random variables in the class-level model, which is relatively small. For instance, the Bayes net of Figure 1 contains four class-level random variables. This allows a query like

$$P(\mathit{gender}(X) | \mathit{gender}(Y), \mathit{Friend}(X, Y)),$$

which predicts the gender of a single generic user, given the gender of a single generic friend. Halpern [3] investigates connections between instance-level probabilities and class-level probabilities. Schulte [27] discusses the importance of such connections for statistical-relational learning.

Statistical Relational Models. To our knowledge, the Statistical Relational Models (SRMs) of Getoor, Taskar and Koller [8], are the only prior statistical models with a class-level probability semantics. A direct empirical comparison is difficult as code has not been released, but SRMs have compared favorably with benchmark methods for database cardinality estimation [2].

SRMs differ from FBNs and other statistical-relational models in several respects. (1) SRMs are derived from a tuple semantics [8, Def.6.3] which is different from the random selection semantics we propose for FBNs. (2) SRMs are less expressive: They cannot express general combinations of positive and negative relationships [8, Def.6.11]. This restriction stems from the fact that the SRM semantics is based on randomly selecting tuples from *existing tables* in the database. Complements of relationship tables are usually not stored (e.g., there is no table that lists the set of user pairs who are *not* friends). The expressive power of SRMs and FBNs becomes essentially equivalent if the SRM semantics is extended to include drawing random tuples from complement tables, but this entails the large increase in storage and processing described above.

The Complete-Data and Closed-World Assumptions. In logic, the closed-world assumption is that "atomic sentences not known to be true are in fact false" [13, Ch.8.2.8]. While the closed-world assumption is used in logical reasoning rather than learning, it is relevant for learning if we view it as *an assumption about how the data are generated*. Specifically, we assume that if a link between two individuals is not listed in a database, it does not exist (cf. Figure 1). To illustrate, we may assume that if a student registers in a course, this event is recorded in the university database, so the absence of a record implies that the student has not registered. If this assumption is not warranted for a particular data generating mechanism, the complete-data assumption fails, because for some pairs of individuals, the data does not indicate whether a link between does not actually exist or simply has not been recorded. One way to deal with missing link data is to use imputation methods [31, 32]. Another approach is downsampling existing links by adding random negative links [18, 33]. The important point for this paper is that our methods can be used no matter how link frequencies are estimated, whether from corrected or uncorrected observed link counts. Specifically, given estimates for the Möbius parameters, the Möbius transform finds the probabilities for negative link events that those parameters determine. This computation assumes nothing but the laws of the probability calculus.

Parfactors and Lifted Inference. Lifted inference aims to speed up instance-level inferences by computing *parfactors*, that is, counts of equivalent events, rather than repeating equivalent computations [7]. Based on the similarity with computing event counts in a database, Schulte and Khosravi refer to learning with the pseudo-likelihood as *lifted learning* [34]. The counting algorithms in this paper can likely be applied to computing parfactors that involve negated relations. The motivation for parfactors, however, is as a means to the end of computing instance-level predictions (e.g., predict the gender of Sam). In contrast, our motivation is to learn class frequencies as an end in itself. Because parfactors are used with instance-level inferences, they usually concern classes defined with reference to specific individuals, whereas the statistics we model in this paper are at the class-level only.

8 Conclusion

Class-level probabilities represent relational statistics about the frequencies or rates of generic events and patterns. Representing these probabilities has been a long-standing concern of AI research. Class-level probabilities exhibit informative statistical dependencies in a relational structure, and they support applications like strategic planning and query optimization. Using grounding-free semantics and the Inverse Möbious transform for learning the joint probabilities, parametrized Bayes nets can support efficient, accurate learning and inference of class-level probabilities. The class-level interpretation of Bayes nets is based on the classic random selection semantics for probability logic. Parameters are learned using the maxima of a recent pseudo-likelihood function as estimates, which are the empirical frequencies. The inverse Möbius transform makes the computation of database frequencies feasible even when the frequencies involve negated links. In evaluation on four benchmark databases, the maximum pseudo-likelihood estimates approach the true conditional probabilities as observations increase. The fit is good even for medium data sizes. Overall, our simulations show that Bayes net models derived with maximum pseudo-likelihood parameter estimates provide excellent estimates of class-level probabilitie and should be applicable across a wide range of applications.

References

[1] Schulte, O.: A tractable pseudo-likelihood function for Bayes nets applied to relational data. In: SIAM SDM. (2011) 462–473

[2] Getoor, L., Taskar, B., Koller, D.: Selectivity estimation using probabilistic models. ACM SIGMOD Record **30**(2) (2001) 461–472

[3] Halpern, J.Y.: An analysis of first-order logics of probability. Artificial Intelligence **46**(3) (1990) 311–350

[4] Bacchus, F.: Representing and reasoning with probabilistic knowledge: a logical approach to probabilities. MIT Press, Cambridge, MA, USA (1990)

[5] Maier, M., Taylor, B., Oktay, H., Jensen, D.: Learning causal models of relational domains. In: AAAI. (2010)

[6] Babcock, B., Chaudhuri, S.: Towards a robust query optimizer: a principled and practical approach. In: Proceedings of the 2005 ACM SIGMOD international conference on Management of data. SIGMOD '05, New York, NY, USA, ACM (2005) 119–130

[7] Poole, D.: First-order probabilistic inference. In: IJCAI. (2003) 985–991

[8] Getoor, L.: Learning Statistical Models From Relational Data. PhD thesis, Department of Computer Science, Stanford University (2001)

[9] Kennes, R., Smets, P.: Computational aspects of the Möbius transformation. In: UAI. (1990) 401–416

[10] Getoor, L., Friedman, N., Koller, D., Pfeffer, A., Taskar, B.: Probabilistic relational models. [35] chapter 5 129–173

[11] Chiang, M., Poole, D.: Reference classes and relational learning. Int. J. Approx. Reasoning **53**(3) (2012) 326–346

[12] Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D., Kolobov, A.: BLOG: Probabilistic models with unknown objects. In: Statistical Relational Learning. MIT Press (2007) 373–395

[13] Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach. Prentice Hall (2010)

[14] Ullman, J.D.: Principles of database systems. 2. Computer Science Press (1982)

[15] Cussens, J.: Logic-based formalisms for statistical relational learning. [35]

[16] Bacchus, F., Grove, A.J., Koller, D., Halpern, J.Y.: From statistics to beliefs. In: AAAI. (1992) 602–608

[17] Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan and Claypool Publishers (2009)

[18] Khot, T., Natarajan, S., Kersting, K., Shavlik, J.W.: Learning Markov logic networks via functional gradient boosting. In: ICDM. (2011) 320–329

[19] Yin, X., Han, J., Yang, J., Yu, P.S.: Crossmine: Efficient classification across multiple database relations. In: Constraint-Based Mining and Inductive Databases. (2004) 172–195

[20] Lauritzen, S.L.: Graphical Models (Oxford Statistical Science Series). Oxford University Press, USA (July 1996)

[21] Domingos, P., Richardson, M.: Markov logic: A unifying framework for statistical relational learning. [35]

[22] Khosravi, H., Man, T., Hu, J., Gao, E., Schulte, O.: Learn and join algorithm code. URL = `http://www.cs.sfu.ca/~oschulte/jbn/`.

[23] Khosravi, H., Schulte, O., Man, T., Xu, X., Bina, B.: Structure learning for Markov logic networks with many descriptive attributes. In: AAAI. (2010) 487–493

[24] The Tetrad Group: The Tetrad project (2008) http://www.phil.cmu.edu/projects/tetrad/.

[25] Allen, T.V., Singh, A., Greiner, R., Hooper, P.: Quantifying the uncertainty of a belief net response: Bayesian error-bars for belief net inference. Artif. Intell. **172**(4-5) (2008) 483–513

[26] Kok, S., Summer, M., Richardson, M., Singla, P., Poon, H., Lowd, D., Wang, J., Domingos, P.: The Alchemy system for statistical relational AI. Technical report, University of Washington. (2009) Version 30.

[27] Schulte, O.: Challenge paper: Marginal probabilities for instances and classes. ICML-SRL Workshop on Statistical Relational Learning. (June 2012)

[28] Kok, S., Domingos, P.: Learning Markov logic networks using structural motifs. In: ICML. (2010) 551–558

[29] Frank, O.: Estimation of graph totals. Scandinavian Journal of Statistics **4:2** (1977) 81–89

[30] Getoor, L., Taskar, B.: Introduction. [35] 1–8

[31] Hoff, P.D.: Multiplicative latent factor models for description and prediction of social networks. Computational and Mathematical Organization Theory (2007)

[32] Namata, G.M., Kok, S., Getoor, L.: Collective graph identification. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining. KDD '11, New York, NY, USA, ACM (2011) 87–95

[33] Yang, S.H., Long, B., Smola, A., Sadagopan, N., Zheng, Z., Zha, H.: Like like alike: joint friendship and interest propagation in social networks. In: Proceedings of the 20th international conference on World wide web. WWW '11, New York, NY, USA, ACM (2011) 537–546

[34] Schulte, O., Khosravi, H.: Learning graphical models for relational data via lattice search. Machine Learning **88:3** (2012) 331–368

[35] Getoor, L., Taskar, B.: Introduction to statistical relational learning. MIT Press (2007)