Fast Parameter Learning for Markov Logic Networks Using Bayes Nets

Hassan Khosravi
hkhosrav@cs.sfu.ca
School of Computing Science
Simon Fraser University
Vancouver-Burnaby, B.C., Canada

Abstract

Markov Logic Networks (MLNs) are a prominent statistical relational model that consist of weighted first order clauses. Structure and parameter learning for MLNs is a challenging active area of current research. For complex relational schemas with many descriptive attributes, one of the most effective algorithms is the moralization approach: Use any Bayes net algorithm to learn a directed relational graphical model, then convert to first order clauses. While this is a fast structure learning method, optimizing parameters for a moralized Bayes net using standard MLN techniques is a bottleneck. In this paper we investigate a moralization approach to parameter estimation where MLN weights are directly inferred from Bayes net parameters. Empirical evaluation indicates that parameter estimation via moralization is orders of magnitude faster than parameter optimization, while performing as well or better on prediction metrics.

1 Introduction

Statistical relational learning (SRL) concerns the induction of probabilistic knowledge that supports accurate prediction for multi-relational structured data [1]. Markov Logic Networks (MLNs) form one of the most prominent SRL model classes; they generalize both first-order logic and Markov network models [2]. An MLN is represented as a set of weighted clauses in first-order logic. Learning an MLN decomposes into structure learning, learning the logical clauses, and parameter learning, setting the weight of each clause. MLNs have become popular as a unifying framework for SRL [3, 4]. An open-source benchmark system for MLNs is the Alchemy package [5]. Most previous learning methods for both structure learning and parameter learning are still inefficient and only applicable to small-to-medium datasets only as big as 2,673 ground atoms in [6, 7] and upto 42558 ground atoms in [8].

The learn-and-join algorithm [9, 10] applied Bayes net (BN) algorithms to perform structure learning for MLNs in relational datasets with schemas that feature a significant number of descriptive attributes, compared to the number of relationships. The evaluation was done on datasets with up to 170,000 true ground atoms. Khosravi *et al.* used moralization method to produce MLN clauses from the BN structure and then applied MLN weight learning techniques to the moralized clauses [9, 10]. Weight learning in Markov logic is a convex optimization problem, and thus gradient descent is guaranteed to find the global optimum. However, convergence to this optimum may be extremely slow, partly because the problem is ill-conditioned since different clauses may have very different numbers of satisfying groundings [11, 2]. In the experiments of Khosravi *et al.*, over 95% of MLN learning time was spent on parameter estimation, which sometimes even exceeded system resources. In this paper we propose a new *parameter moralization* method for MLN weight estimation.

Approach. A parameter moralization method estimates parameters from the database statistics using fast Bayes net and decision tree algorithms. Then a *conversion function* maps the conditional probabilities into MLN weights. Domingos and Richardson proposed using the log-cps,logarithm of conditional probabilities, as clause weights [2]. We illustrate situations regarding ill-conditioning in which this method does not work well and introduce a new method of conversion, LOG-LINEAR, that leads to theoretically more sound and empirically more accurate predictions.

We evaluated our learning algorithms using cross-validation on five well known public domain datasets. Parameter moralization methods are orders of magnitude faster than local optimization techniques, while their predictive accuracy is competitive or even superior.

Paper Organization. We discuss related work, background, and notation. We present a method for estimating relational conditional probabilities using BN algorithms and database techniques. Conversion functions for mapping conditional probabilities into weights are introduced. We compare our approach both in terms of processing speed and in terms of model accuracy with other state of the art MLN learning algorithms.

1.1 Related work

Most work on parameter learning in MLNs is based on ideas developed for Markov networks (undirected graphical models) in the propositional case. Special issues that arise with relational data are discussed by Lowd and Domingos [11]. Most recent methods aim to maximize the regularized weighted pseudo log-likelihood [2, 8], and/or perform a scaled conjugate gradient descent using second-order derivative information [11]. Our approach utilizes parameter estimation algorithms from directed rather than undirected graphical models. As far as we know, our work is the first on MLN learning that uses a relational database for data management, which allows it to take advantage of efficient database querying techniques that are less affected by the number and arity of the descriptive attributes in a database.

The main motivation for converting the directed model into an undirected model and performing inference with an undirected model is that they do not suffer from the problem of cyclic dependencies in relational data [2, 12, 9]. Early work on this topic required ground graphs to be acyclic [13, 14]. For example, Probabilistic Relational Models allow dependencies that are cyclic at the predicate level as long as the user guarantees acyclicity at the ground level [14]. A recursive dependency of an attribute on itself is shown as a self loop in the model graph. If there is a natural ordering of the ground atoms in the domain (e.g., temporal), there may not be cycles in the ground graph; but this assumption is restrictive in general. The generalized order-search of Ramon *et al.* [15] instead resolves cycles by learning an ordering of ground atoms which complicates the learning procedure. Our approach combines the scalability and efficiency of directed model search, and the inference power and theoretical foundations of undirected relational models.

2 Background Concepts

A **Bayes net structure** [16] is a directed acyclic graph (DAG) G, whose nodes comprise a set of random variables denoted by V. A Bayes net (BN) is a pair $\langle G, \boldsymbol{\theta}_G \rangle$ where $\boldsymbol{\theta}_G$ is a set of parameter values that specify the probability distributions of children conditional on assignments of values to their parents. We use as our basic model class **Parametrized Bayes Nets** [17], a relatively straightforward generalization of Bayes Nets for relational data. Our methods also apply to other directed graphical formalisms. A **population** is a set of individuals, corresponding to a domain or type in logic. A **parametrized random variable** (PRV) is of the form $f(t_1, \ldots, t_k)$ where f is a **functor** (either a function symbol or a predicate symbol) and each t_i is a first-order variable or a constant. Each functor has a set of values (constants) called the **range** of the functor. A **Parametrized Bayes Net structure** consists of: (1) A directed acyclic graph (DAG) whose nodes are parametrized random variables. (2) A population for each first-order variable. (3) An assignment of a range to each functor. A **Parametrized Bayes Net** (PBN) is a BN whose graph is a PBN structure.[1]

Relational Schemas. We assume a standard relational schema containing a set of tables, each with key fields, descriptive attributes, and foreign key pointers. A **database instance** specifies the tuples contained in the tables of a given database schema. A **table join** of two or more tables contains the rows in the Cartesian products of the tables whose values match on common fields.

Markov Logic Networks are presented in detail by Domingos and Richardson [2]. The qualitative component or structure of an MLN is a finite set of 1st-order formulas or clauses $\{p_i\}$, and its quantitative component is a set of weights $\{w_i\}$, one for each formula.

Moralized Bayes Nets The learn-and-join algorithm applied Bayes net (BN) algorithms to perform structure learning for MLNs in relational datasets with schemas that feature a significant number of descriptive attributes [9]. **Moralization** is a technique used to convert a directed acyclic graph (DAG) into undirected models or MLN formulas. To convert a Bayes net into an MLN using moralization, add a formula to the MLN for each assignment of values to a child and its parents [2, Sec. 12.5.3]. The MLN for moralized BN B thus contains a formula for each CP-table entry in B [2] which we call **MBN** structure in this paper. Figure 1 illustrates a Parameterized Bayes net learned using the learn-and-join algorithm and Figure 2(a) shows the conditional probability table and its corresponding clauses for the node ranking.

[1] The term "Parametrized" refers to the semantics of PBNs, and does not mean that parameters have been assigned for the PBN structure.

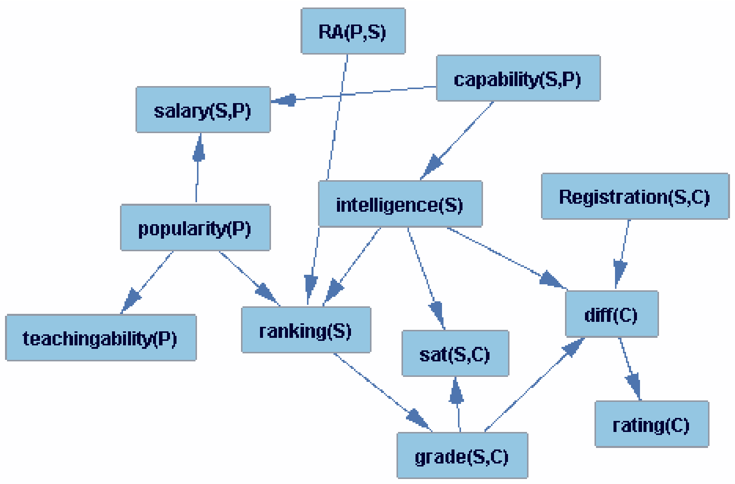


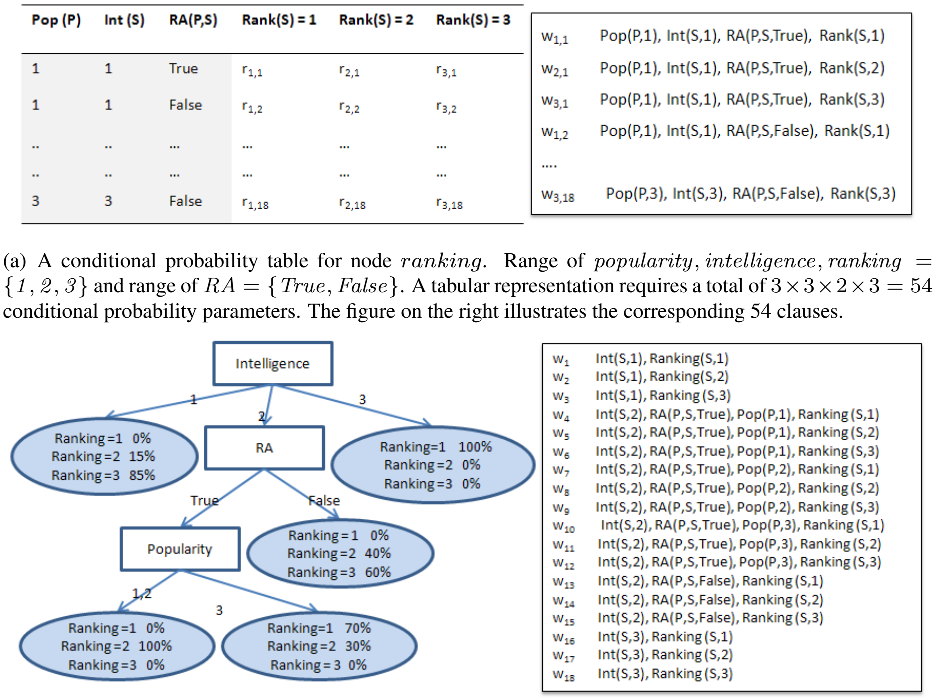
Figure 1: A parametrized Bayes net graph.

While the moralization approach produces graph structures that represent the dependencies among predicates well, converting each row of each conditional probability table to an MLN clause leads to a large number of MLN clauses and hence MLN parameters.

Local or **context-sensitive** independencies are a well-known phenomenon that can be exploited to reduce the number of parameters required in a Bayes net. A **decision tree** can compactly represent conditional probabilities [18]. The nodes in a decision tree for a Parametrized RV c are parametrized random variables. An edge that originates in a PRV $f(t_1, \dots, t_k)$ is labeled with one of the possible values in the range of f. The leaves are labeled with probabilities for the different possible values of the c variable. Khosravi et al combine decision tree learning algorithms with Bayes nets to learn a compact set of clauses for relational data [10]. We call this structure **MBN-DT** in this paper. In their experiments, using the decision tree representation (i.e., MBN-DT) instead of "flat" conditional probability tables (i.e., MBN) reduced the number of MLN weight parameters by a factor of 5-25. Figure 2(b) shows the decision tree and its corresponding MLN clauses for the ranking node in 1.

3 Relational Parameter Learning for Bayes Nets

We assume that a structure of a Parameterized Bayes net, a relational database, and a method for storing the parameters as either conditional probability tables or decision trees are given as input. The algorithm estimates parameter for the given structure as output. The structure may have been obtained from various relational structure learning methods, such as those for Probabilistic Relational Models [19], the learn-and-Join algorithm [9], and RelationalPC [20]. Our basic approach is to consider each family in



Pop (P)	Int (S)	RA(P,S)	Rank(S) = 1	Rank(S) = 2	Rank(S) = 3
1	1	True	$r_{1,1}$	$r_{2,1}$	$r_{3,1}$
1	1	False	$r_{1,2}$	$r_{2,2}$	$r_{3,2}$
..	..	...	...	...	...
..	..	...	...	...	...
3	3	False	$r_{1,18}$	$r_{2,18}$	$r_{3,18}$

$w_{1,1}$ Pop(P,1), Int(S,1), RA(P,S,True), Rank(S,1)
$w_{2,1}$ Pop(P,1), Int(S,1), RA(P,S,True), Rank(S,2)
$w_{3,1}$ Pop(P,1), Int(S,1), RA(P,S,True), Rank(S,3)
$w_{1,2}$ Pop(P,1), Int(S,1), RA(P,S,False), Rank(S,1)
....
$w_{3,18}$ Pop(P,3), Int(S,3), RA(P,S,False), Rank(S,3)

(a) A conditional probability table for node *ranking*. Range of *popularity*, *intelligence*, *ranking* $= \{1, 2, 3\}$ and range of $RA = \{True, False\}$. A tabular representation requires a total of $3 \times 3 \times 2 \times 3 = 54$ conditional probability parameters. The figure on the right illustrates the corresponding 54 clauses.

w_1 Int(S,1), Ranking(S,1)
w_2 Int(S,1), Ranking(S,2)
w_3 Int(S,1), Ranking (S,3)
w_4 Int(S,2), RA(P,S,True), Pop(P,1), Ranking (S,1)
w_5 Int(S,2), RA(P,S,True), Pop(P,1), Ranking (S,2)
w_6 Int(S,2), RA(P,S,True), Pop(P,1), Ranking (S,3)
w_7 Int(S,2), RA(P,S,True), Pop(P,2), Ranking (S,1)
w_8 Int(S,2), RA(P,S,True), Pop(P,2), Ranking (S,2)
w_9 Int(S,2), RA(P,S,True), Pop(P,2), Ranking (S,3)
w_{10} Int(S,2), RA(P,S,True), Pop(P,3), Ranking (S,1)
w_{11} Int(S,2), RA(P,S,True), Pop(P,3), Ranking (S,2)
w_{12} Int(S,2), RA(P,S,True), Pop(P,3), Ranking (S,3)
w_{13} Int(S,2), RA(P,S,False), Ranking (S,1)
w_{14} Int(S,2), RA(P,S,False), Ranking (S,2)
w_{15} Int(S,2), RA(P,S,False), Ranking (S,3)
w_{16} Int(S,3), Ranking (S,1)
w_{17} Int(S,3), Ranking (S,2)
w_{18} Int(S,3), Ranking (S,3)

(b) A decision tree that specifies conditional probabilities for the $ranking(S)$ node in Figure 1 and the corresponding MLN clauses generated from the decision tree.

Figure 2: The MLN structure generated from MBNs [9] and MBN-DT [10]

the structure (node + parents), form a **family data table**, and apply any propositional Bayes Net or Decision Tree parameter estimation algorithm to the family structure and data table. The family data table is defined by joining the minimum subset of tables that contains all the attributes and relationships that occur in the family. We select (project) only the attributes that occur in the family. The family data table specifies the *sufficient statistics*, that is, the number of satisfying groundings for each assignment of values to the family nodes. Algorithm 1 summarizes the procedure with pseudocode.

For example, in the PBN of Figure 1, the family of $\mathit{diff}(C)$ comprises in addition the parent nodes $grade(S, C)$, $intelligence(S)$, $Registration(S, C)$. Thus the family data table is given by the join $Registration \bowtie Student \bowtie Course$. followed by selecting (projecting) the attributes $intelligence, \mathit{difficulty}, grade$. A propositional Bayes net or decision tree parameter learner is applied to the resulting table with the family graph, that contains $\mathit{difficulty}(\mathbf{C})$ and its parents. The general approach of applying a propositional learners to a relational data table was also followed in the Learn-and-Join structure learning method. The table joins can be efficiently computed with SQL queries.

This method implicitly considers only links or link chains that exist in the database (true relationship groundings). It can extended for conditional probabilities that involve non-existing relationships. The main problem in this case is computing sufficient database statistics (frequencies), which can be addressed with the dynamic programming algorithm of Khosravi *et al.* [21]. Experimentally we found that including information from non-existing links is not helpful for predicting the attribute of a target entity (link-based classification) because information from unrelated entities tends to be irrelevant, but also tends to carry too much weight because there are typically may more unrelated than relate entities. So when converting the PBN to an MLN, we only include clauses with existing links (i.e., where relationship indicator nodes are true).

Algorithm 1 Conditional Probability Estimation for Parameterized Bayes Nets.

```
Input: Database instance $\mathcal{D}$; DAG $G$ ; Method $M$ [which is either BN or DT]
Output: Parameterized Bayes net $\langle G, \boldsymbol{\theta}_G \rangle$ with $\boldsymbol{\theta}_{G,v}$ for child node $v$.
Call: Find-Datatable($Family$, $\mathcal{D}$). [Outputs a join data table for the nodes in a family (child + parents) from database $\mathcal{D}$]
Call: BPL($T$, $Child$, $Parents$). [Uses any single-table Bayes net Parameter Learner
Call: DTL($T$, $Child$, $Parents$). [Uses any single-table Decision tree Parameter Learner to estimate conditional probabilities for $Child$ given $Parents$ from data table $T$]
for all parametrized random variables $PRV$ do
    $Family$ := Parents($PRV$) $\cup \{PRV\}$
    $T$ := Find-Datatable($Family$, $\mathcal{D}$)
    if M = BN then
        $\theta_{G,PRV}$ = BPL($T$, $PRV$, Parents($PRV$))
    end if
    if M = DT then
        $\theta_{G,PRV}$ = DTL($T$, $PRV$, Parents($PRV$))
    end if
end for
return $\langle G, \boldsymbol{\theta}_G \rangle$
```

4 Conversion Functions

In the following discussion, fix a PBN B and a child node v with k possible values $v_1, \ldots, v_k$ and an assignment π of values to its parents. Then the conditional probability $p(v_i|\pi)$ is defined in the CP-table or decision tree leafs for B. The moralized MBN contains a formula p_i that expresses that child node v takes on value i and the parents take on value π. The weight of formula p_i is denoted as w_i.

In order to convert the conditional probabilities from parameterized Bayes Nets into weights for MLNs, using the logarithm of the conditional probabilities was suggested by Domingos and Richardson [2] as part of the standard moralization procedure. We call this method LOGPROB. The LOGPROB method sets

$$w_i := log(p(v_i|\pi))$$

, which weights events with small conditional probabilities exponentially and ignores the ones with probability one. We use the Laplace correction for events with zero instances in the data.

In the propositional case, combining moralization with the log-conditional probabilities as in the LOGPROB method leads to an undirected graphical model that is predictively equivalent to the original directed graphical model [16]. Although there is no corresponding result for the case of relational data, the propositional conversion result makes the log-probabilities a plausible candidate for weights in a moralized 1st-order Bayes net. Theoretical support for the LOGPROB method is provided by considering the log-likelihood function for the moralized MBN structure. The standard log-likelihood for an MLN M [2] is given by

$$L_M(\mathcal{D}) = \sum_j w_j n_j(\mathcal{D}) + ln(Z)$$

where $n_j(\mathcal{D})$ denotes the number of instances of formula j in database $\mathcal{D}$ and Z is a normalization constant. Omitting the normalization term $ln(Z)$, for moralized Bayes nets this is the sum, over all child-parent configurations, of the Bayes net log conditional probability of the child given the parent, multiplied by $n_{j,\pi}(\mathcal{D})$, which is the number of instances of the child-parent configuration in the database;

$$L_M(\mathcal{D}) = \sum_i \sum_\pi \log(p(v_i|\pi))n_{i,\pi}(\mathcal{D})$$

This unnormalized log-likelihood is maximized by using the observed conditional frequencies in the database. (The argument is exactly analogous to maximum likelihood estimation for a single data table). While the normalization constant is required for defining valid probabilistic inferences, it arguably does not contribute to measuring the fit of a parameter setting and hence can be ignored in model selection; the constraint that weights are derived from normalized conditional probabilities in a Bayes net already bounds their range.

The LOGPROB method works well with the MBN structure but performs very poorly with the MBN-DT structure. The issue that arises with the MBN-DT method is that different formulas may have very different number of groundings, depending on the predicates and 1st-order variables they contain. To illustrate the issue, consider again Figures 2(a) and 2(b). Let us focus on the $\mathit{Intelligence}$ predicate. When the Markov Logic inference model evaluates the probability that $\mathit{Intelligence}$ of a particular target entity s is 1, it considers only one of the short formulas of weight w_1, w_2, or w_3. These formulas involve only the $\mathit{Intelligence}$ and $\mathit{Ranking}$ predicates, which will have only 1 grounding for a fixed target entity s. (Each student has exactly one ranking). Thus for the assignment $\mathit{Intelligence} = \mathit{1}$, only one grounding is relevant.

Now consider the assignment $\mathit{Intelligence} = \mathit{2}$. If we restrict attention to formulas in which $RA(P, S)$ is true (i.e., professors P such that student s is an RA for P), there are 9 formulas with weights $w_4 - w_{12}$ These formula will have more than one grounding; for instance, if the target entity s was an RA for 10 professors, the total number of groundings is 10. In general, the number of groundings in the larger formulas will be at least as great as the number of linked entities. Now log-probabilities are negative, so using the LOGPROB method means that relatively many negative weights will be added up in evaluating the assignment $Intelligence = \mathit{2}$ compared to the assignment $Intelligence = \mathit{1}$. Thus the LOGPROB method leads to a bias towards $Intelligence = \mathit{1}$. This illustrates how LOGPROB induces a bias against values that satisfy formulas with more groundings. Note that, in the standard moralization method based on CP-tables, as shown in Figure 2(a), the bias of the LOGPROB does not occur because all values appear in the same number of the rules with similar structure and hence groundings.

We propose another LOG-LINEAR conversion method which works well with both MBN and MBN-DT structure. LOG-LINEAR sets

$$w_i := log(p(v_i|\pi)) - log(1/k)$$

Weights set in this way can be seen as measuring the information gain provided by the parent information π relative to the uniform probability baseline. These weights can be interpreted as usual in a linear model: A positive weight indicates that a predictive factor increases the baseline probability, a negative weight indicates a decreased probability relative to the baseline. A zero weight indicates a condition that is irrelevant in the sense of not changing the baseline probability.

With the LOG-LINEAR transformation, some of the formulas receive positive and some negative weights, so there is no bias against values that are involved in formulas with more groundings. That is, the influences of the different groundings are more balanced against each other. For instance, if the ranking of the target student is 2, then all instances of professors P with popularity 3 and $RA(P, S)$ true contribute the negative weight $ln(30\%) - ln(33\%)$. In contrast, all instances of professors P with popularity 1 or 2 and $RA(P, S)$ true contribute the positive weight $ln(100\%) - ln(33\%)$.

5 Experimental Design

We first discuss the datasets used, then the systems compared, finally the comparison metrics.

5.1 Datasets

We used five benchmark real-world datasets. Table 1 lists the resulting databases and their sizes in terms of total number of tuples and number of ground atoms, which is the input format for Alchemy.

Each descriptive attribute is represented as a separate function, so the number of ground atoms is larger than that of tuples.

MovieLens Database. The first dataset is the MovieLens dataset from the UC Irvine machine learning repository. [9].

Mutagenesis Database. This dataset is widely used in ILP research [22]. It contains information on Atoms, Molecules, and Bonds between them.

Hepatitis Database. This data is a modified version of the PKDD02 Discovery Challenge database, following [23]. The database contains information on the laboratory examinations of hepatitis B and C infected patients.

Mondial Database. This dataset contains data from multiple geographical web data sources. We follow the modification of [24], and use a subset of the tables and features. Our dataset includes a self-relationship table *Borders* that relates two countries.

UW-CSE database. This dataset lists facts about the Department of Computer Science and Engineering at the University of Washington (UW-CSE) (e.g., Student, Professor) and their relationships (i.e. AdvisedBy, Publication). The dataset was obtained by crawling pages in the department's Web site (www.cs.washington.edu).

Dataset	**#tuples**	**#Ground atoms**
Movielens	82623	170143
Mutagenesis	15218	35973
Hepatitis	12447	71597
Mondial	814	3366
UW-CSE	2099	3380

Table 1: Size of datasets in total number of table tuples and ground atoms.

5.2 Comparison Systems and Performance Metrics.

Structure learning. We fix the structure for all the methods to evaluate just the parameters. We use two different structure learning methods to evaluate the parameter learning methods with both dense and sparse structures. We used the **MBN** [9] to get a dense structure with conditional probabilities and **MBN-DT** [10] to get a sparse structure with Decision trees. Both methods use GES search [25] and the BDeu score as implemented in version 4.3.9-0 of CMU's Tetrad package (structure prior uniform, ESS=10; [26]).

Our experiments compare the following methods parameter learning methods.

MLN. Weight learning is carried out with the procedure of Lowd and Domingos [11, 3] , implemented in Alchemy.

LSM Learning Structural Motifs (LSM; [8]) uses random walks to identify densely connected objects in data, and groups them and their associated relations into a motif. We input the structure of the learn-and-join algorithms to LSM. Running LSMs structure learning algorithm tries to prune the structure.

LOGPROB Weight learning is carried out using the algorithm discussed in Section 3. Parameters are given by Tetrad's maximum likelihood estimation method and the LOGPROB conversion.

LOG-LINEAR weight learning is the same as LOGPROB method but we use the LOG-LINEAR conversion.

We report measurements on Runtime and Accuracy. To define accuracy, we apply MLN inference to predict the probability of an attribute value, and score the prediction as correct if the most probable value is the true one. For example, to predict the gender of person Bob, we apply MLN inference to the atoms gender(Bob, male) and gender(Bob, female). The result is correct if the predicted probability of gender(Bob, male) is greater than that of gender(Bob, female). The values we report are averages over all attribute predicates.

Infernce. We use the MC-SAT inference algorithm [27] implemented in Alchemy to compute a probability estimate for each possible value of a descriptive attribute for a given object or tuple of objects

6 Evaluation Results

We discuss run time and then accuracy. We investigated the predictive performance by doing five-fold cross validation on the given datasets. All experiments were done on a QUAD CPU Q6700 with a 2.66GHz CPU and 8GB of RAM.

6.1 Run Times.

Table 2 shows the time taken in seconds for learning the parameters for Markov Logic Networks using the structures generated by MBN and MBN-DT. The time for the conversion methods is basically the same, namely the time required to compute the database statistics for the entries. For the purposes of discussing runtime, we group LOGPROB and LOG-LINEAR methods and call it Log/Lin in this table. The runtime improvements of orders of magnitude that result from extending the moralization approach to parameter learning findings provide strong evidence that the moralization approach leverages the scalability of relational databases for data management and Bayes nets learning to achieve scalable MLN learning on databases of realistic size—for both structure and parameter learning.

Table 2: The time taken in seconds for parameter learning. we group LOGPROB and LOG-LINEAR methods and call it Log/Lin in this table.

Structure Learning	MBN			MBN-DT		
Parameter Learning	Log/Lin	MLN	LSM	Log/Lin	MLN	LSM
UW-CSE	2	5	80	3	3	8
Mondial	3	90	260	3	15	26
MovieLens	8	10800	14300	9	1800	2100
Mutagenesis	3	9000	58000	4	600	1200
Hepatitis	3	23000	34200	5	4000	5000

6.2 Accuracy.

Table 3 and Table 4 shows the accuracy results using the MBN and MBN-DT respectively. Higher numbers indicate better performance. For the MBN structure,The LOGPROB, LOG-LINEAR, and MLN methods are competitive. LSM clearly performs worse. For the MBN-DT structure, the LOGPROB method performs very poorly as discussed before. The LOG-LINEAR and MLN methods are competitive and have performance as well as each other.

	LOGPROB	LOG-LINEAR	MLN	LSM
UW-CSE	0.72 ± 0.083	0.76 ± 0.022	0.75 ± 0.028	0.64 ± 0.086
Mondial	0.40 ± 0.060	0.41 ± 0.045	0.44 ± 0.050	0.32 ± 0.042
Movielens	0.64 ± 0.006	0.64 ± 0.006	0.60 ± 0.029	0.57 ± 0.016
Mutagenesis	0.55 ± 0.139	0.64 ± 0.025	0.61 ± 0.022	0.64 ± 0.029
Hepatitis	0.49 ± 0.033	0.50 ± 0.037	0.51 ± 0.025	0.30 ± 0.028

Table 3: The 5-fold cross-validation estimate using MBN structure learning for the accuracy of predicting the true values of descriptive attributes, averaged over all descriptive attribute instances. Observed standard deviations are shown.

	LOGPROB	LOG-LINEAR	MLN	LSM
UW-CSE	0.06 ± 0.088	0.73 ± 0.166	0.75 ± 0.086	0.65 ± 0.076
Mondial	0.18 ± 0.036	0.43 ± 0.027	0.44 ± 0.033	0.31 ± 0.024
Movielens	0.26 ± 0.017	0.62 ± 0.026	0.62 ± 0.023	0.59 ± 0.051
Mutagenesis	0.21 ± 0.021	0.61 ± 0.023	0.60 ± 0.027	0.61 ± 0.025
Hepatitis	0.19 ± 0.024	0.48 ± 0.032	0.50 ± 0.021	0.40 ± 0.032

Table 4: The 5-fold cross-validation estimate using MBN-DT structure learning for the accuracy of predicting the true values of descriptive attributes, averaged over all descriptive attribute instances. Observed standard deviations are shown.

7 Conclusion and Future Work

This paper considered the task of building a statistical-relational model for databases with many descriptive attributes. The moralization approach combines Bayes net learning, one of the most successful machine learning techniques, with Markov Logic networks, one of the most successful statistical-relational formalisms. Previous work applied the moralization method to learning MLN structure; in this paper we extended it to learning MLN parameters as well. We motivated and empirically investigated a new method for converting Bayes net paramters to MLN weights. Our evaluation on five medium-size benchmark databases with descriptive attributes indicates that compared to previous MLN learning methods, the moralization parameter learning approach improves the scalability and run-time performance by at least two orders of magnitude. Predictive accuracy is competitive or even superior.

References

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

[2] Domingos, P., Richardson, M.: Markov logic: A unifying framework for statistical relational learning. [1]

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

[4] Mapping and Revising Markov Logic Networks for Transfer Learning. In: AAAI. (2007)

[5] 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.

[6] Mihalkova, L., Mooney, R.J.: Bottom-up learning of Markov logic network structure. In: ICML, ACM (2007) 625–632

[7] Kok, S., Domingos, P.: Learning markov logic network structure via hypergraph lifting. In: ICML. (2009) 64–71

[8] Kok, S., Domingos, P.: Learning markov logic networks using structural motifs. In Fürnkranz, J., Joachims, T., eds.: ICML, Omnipress (2010) 551–558

[9] 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

[10] Khosravi, H., Schulte, O., Hu, J., Gao, T.: Learning compact markov logic networks with decision trees. In: ILP. (2011)

[11] Lowd, D., Domingos, P.: Efficient weight learning for Markov logic networks. In: PKDD. (2007) 200–211

[12] Taskar, B., Abbeel, P., Koller, D.: Discriminative probabilistic models for relational data. In: UAI. (2002) 485–492

[13] Kersting, K., de Raedt, L.: Bayesian logic programming: Theory and tool. [1] chapter 10 291–318

[14] Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: In IJCAI, Springer-Verlag (1999) 1300–1309

[15] Ramon, J., Croonenborghs, T., Fierens, D., Blockeel, H., Bruynooghe, M.: Generalized ordering-search for learning directed probabilistic logical models. Machine Learning **70** (2008) 169–188

[16] Pearl, J.: Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann (1988)

[17] Poole, D.: First-order probabilistic inference. In Gottlob, G., Walsh, T., eds.: IJCAI, Morgan Kaufmann (2003) 985–991

[18] Boutilier, C., Friedman, N., Goldszmidt, M., Koller, D.: Context-specific independence in bayesian networks. In: Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence, Citeseer (1996) 115–123

[19] Getoor, L., Friedman, N., Koller, D., Pfeffer, A., Taskar, B.: Probabilistic relational models. [1] chapter 5 129–173

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

[21] Khosravi, H., Schulte, O., Bina, B.: Virtual joins with nonexistent links, 19th Conference on Inductive Logic Programming (ILP) (2009) URL = `http://www.cs.kuleuven.be/~dtai/ilp-mlg-srl/papers/ILP09-39.pdf`.

[22] Srinivasan, A., Muggleton, S., Sternberg, M., King, R.: Theories for mutagenicity: A study in first-order and feature-based induction. Artificial Intelligence **85** (1996) 277–299

[23] Frank, R., Moser, F., Ester, M.: A method for multi-relational classification using single and multi-feature aggregation functions. In: PKDD. 430-437 (2007)

[24] She, R., Wang, K., Xu, Y.: Pushing feature selection ahead of join. In: SIAM SDM. (2005)

[25] Chickering, D.: Optimal structure identification with greedy search. Journal of Machine Learning Research **3** (2003) 507–554

[26] The Tetrad Group, Department of Philosophy, C.: The Tetrad project: Causal models and statistical data (2008) http://www.phil.cmu.edu/projects/tetrad/.

[27] Poon, H., Domingos, P.: Sound and efficient inference with probabilistic and deterministic dependencies. In: AAAI, AAAI Press (2006)