A Survey on Statistical Relational Learning

Hassan Khosravi and Bahareh Bina

School of Computing Science, Simon Fraser University,
Burnaby, B.C., Canada V5A 1S6
{hkhosrav,bba18}@cs.sfu.ca

Abstract. The vast majority of work in Machine Learning has focused on propositional data which is assumed to be identically and independently distributed, however, many real world datasets are relational and most real world applications are characterized by the presence of uncertainty and complex relational structure where the data distribution is neither identical nor independent. An emerging research area, Statistical Relational Learning(SRL), attempts to represent, model, and learn in relational domain. Currently, SRL is still at a primitive stage in Canada, which motivates us to conduct this survey as an attempt to raise more attention to this field. Our survey presents a brief introduction to SRL and a comparison with conventional learning approaches. In this survey we review four SRL models(PRMs, MLNs, RDNs, and BLPs) and compare them theoretically with respect to their representation, structure learning, parameter learning, and inference methods. We conclude with a discussion on limitations of current methods.

1 Introduction

The vast majority of work in learning has focused on propositional data which consists of identically structured entities that are assumed to be independent. However, many real world datasets are relational. Relational data consists of different types of entities where each entity is characterized with a different set of attributes. Relational data are more complex and better suited with our surroundings where examples are given as multiple related tables. The structure of relational data provides an opportunity for objects to carry additional information via their links and enables the model to show correlations among objects and their relationships.

Statistical Relational Learning(SRL) is a new branch of machine learning that tries to model a joint distribution over relational data[9]. SRL is a combination of statistical learning which addresses uncertainty in data and relational learning which deals with complex relational structures. A statistical relational model for a given database shows not only the correlations between attributes of each table, but also dependencies among attributes of different tables. Statistical relational models are usually represented with graphical models [22] and are different in methods of representation, learning , and inference.

SRL models have been extensively researched and many applications have been proposed for them[13, 25, 23, 27], but they have lacked popularity among researchers in Canada. Due to this, we try to motivate SRL and their importance in this survey. In Section 2 we cover some background information required for studying this paper and introduce a running example to define some of the new concepts. We emphasize on some

A. Farzindar and V. Keselj (Eds.): Canadian AI 2010, LNAI 6085, pp. 256–268, 2010.

of the main differences of relational data and propositional data which rationalizes the need of new methods for learning on relational data in Section 3. Section 4 shows how propositional and relational learners use graphical models for their representation. We review four of the proposed models in Section 5; the methods of representing, learning, and inference are discussed. Studying current state-of-the-art methods provides a realistic view of the current capabilities for SRL.

2 Background and Notation

Entity-Relationship model. We assume that tables in the relational schema can be divided into *entity tables* and *relationship tables*. This is the case whenever a relational schema is derived from an entity-relationship model (ER model) [24, Ch.2.2]. Symbol E refers to entity tables or objects. Symbol R refers to relationship tables or links. Symbol T refers to generic tables.

To better explain the concepts and demonstrate the notations, we define a running example of a university database, which contains three objects or entity tables: $Student$, $Course$, and $Professor$, and two relationship tables: $Registered$, with foreign key pointers to the $Student$ and $Course$ tables, whose tuples indicate students have registered in which courses, and RA, with foreign key pointers to the $Student$ and $Professor$ tables, whose tuples indicate the RAship of students for professors. Table 1 shows the relational schema of university database. Relationships refer to their related objects using *reference slots*. Each table in the relational database is seen as a class that has some descriptive attributes. A schema is instantiated when actual objects are assigned to each table and the references between them are specified.

Table 1. A relational schema for a University model. Key fields are underlined. The schema has three entities and two relationships.

$Student$($\underline{student\_id}$: integer, $intelligence$: string, $ranking$: string)
$Course$($\underline{course\_id}$: integer, $difficulty$: string, $rating$: string)
$Professor$ ($\underline{professor\_id}$, $teaching\_ability$: string, $popularity$: string)
$Registered$ ($\underline{student\_id}$: integer, $\underline{Course\_id}$: integer, $grade$: string, $satisfaction$: string)
RA ($\underline{student\_id}$: integer, $\underline{professor\_id}$: integer, $salary$: string, $capability$: string)

Graphical models. Graphical models [17, 14] are a popular tool for modeling both statistical models and statistical relational models. They provide a principled approach to dealing with uncertainty and relational data through probability theory. The goal of graphical models is to represent a joint distribution over a set of random variables. A random variable is a pair $X = \langle dom(X), P_X \rangle$ where $dom(X)$ is a set of possible values for X called the **domain** of X and $P_X : dom(X) \rightarrow [0, 1]$ is a probability distribution over these values. We assume in this paper that all random variables have finite domains (i.e., discrete or categorical variables). An **atomic assignment** assigns a value $X = x$ to random variable x, where $x \in dom(X)$. A **joint distribution** P assigns a probability to each conjunction of atomic assignments. The two most common classes of graphical models are Bayesian networks and Markov Networks [22].

A **Bayes net structure** is a directed acyclic graph (DAG) G, whose nodes comprise a set of random variables denoted by X. When discussing a Bayes net, we refer interchangeably to its nodes or its variables. A Bayes net(BN) is a pair $\langle G, \theta_G \rangle$ where θ_G is a set of parameter values that specify the probability distributions of each node conditional on instantiations of their parents. These conditional probabilities are specified in a **conditional probability table** (CP-table) for variable X. A BN $\langle G, \theta_G \rangle$ defines a joint probability distribution over $V = \{v_1, .., v_n\}$. The joint probability of an assignment is obtained by multiplying the conditional probabilities of each node value assignment given its parent value assignments.

A **Markov network** is a model for the joint distribution of a set of variables $X = (X_1, X_2, \ldots X_n)$. It is composed of an undirected graph G and a set of potential functions ϕ_k. The graph has a node for each variable, and the model has a potential function for each clique in the graph. A potential function is a non-negative real-valued function of the state of the corresponding clique. The joint distribution over an assignment $X = x$ represented by a Markov network is given by

$$P(X = x) = \frac{1}{Z} \prod_k \phi_k(x_{\{k\}}) \tag{1}$$

where $(x_{\{k\}})$ is the state of the kth clique. Z, known as the partition function, is given by $Z = \sum_{x \in X} \prod_k \phi_k(x_{\{k\}})$

3 Propositional Data and Relational Data

Most real world data is relational which provides more information about an object via its links; . However, relational data has several major differences to propositional data that makes it more challenging to learn models. Several characteristics of relational data and some of its main differences to propositional data are the following.

1. The representation method for relational data and propositional data is different. A relational database stores data in multiple tables that represent different types of entities and relationships between them. Propositional data is stored in a single table.
2. Propositional data consists of identically structured entities, typically assumed to be independently and identically distributed (iid); however, relational data consists of different entities of different types which may be related to each other. For example, all tuples are of type student in propositional data and we assume all students are independent of each other. Relational data has tuples of different types (students and courses) and they may be statically dependent on each other.
3. In modeling propositional data, the number of different states is exponential in the number of attributes, e.g. with n binary attributes, the number of states is $O(2^n)$. When modeling joint distribution of relational data, the number of different states is exponential in the product of the attributes and objects, e.g. with m objects and n binary attributes in total, the number of states is $O(2^{nm})$.
4. The presence of **autocorrelation** is a feature of relational data which augments the complexity of relational learning. Correlations between values of attributes of

objects of the same type were dubbed autocorrelations by [12]. A relationship between objects of the same entity is required to have autocorrelation. For example, $Friends(s_1, s_2) = true$ is a relationship on entity $Student$ where s_1 and s_2 are friends. There may be a pattern between the intelligence of friends, ie, the attribute value $Intelligence(s_1)$ and the attribute value $Intelligence(s_2)$.

In our example, in order to do classification on a student, statistical relational learners may not only look at the courses that student has taken (i.e. the links of the object itself), but also other students who have taken those courses (i.e. the links of the linked objects).

5. One to many and many to many relationships in which an object is in relation with a set of objects is a character of relational data. For example a student may be registered in a set of courses where the *ranking* of the student is in correlation with the *difficulty* of a set of courses. Dealing with many to many relationships is a challenging problem for many SRL models.

Using relational data, not only leads to more accurate results on traditional tasks like classification and prediction, but also introduces some new tasks on relational data. The most popular tasks introduced on relational data are the following.

- Collective classification [13] is an extension over relational classification in presence of autocorrelation. Relational classification is the task of predicting the class label of an object given its attributes, links. In collective classification, the class labels of the links may be unknown.
- Linked based clustering [25] groups together objects that have similar characteristics based on their own attributes [7] and more importantly, the attributes of their links.
- Link prediction [23] determines whether a relation exists between two objects from the attributes of the objects and their links.

4 Statistical Relational Learners

Due to the underlying complicated multi-table structure and correlations of relational data, statistical learning methods have a different representation to propositional learners. SRL models use three graphical structures in order to fully define their model.

1. Graph $G_D(V_D, E_D)$ is used to show the schema of the database and instances. Nodes of G_D, are the objects of the database, and the edges between them represent the relationships between the objects. A vector is assigned to each node and each edge to keep the information of the attributes of objects and links. This graph presents an instantiated schema of the database used for learning. Figure 1 shows $G_D(V_D, E_D)$ for an instance of Table 1. The graph is not required for propositional learners as the single table provides all the existing information.
2. Graph $G_M(V_M, E_M)$ represents the class model which encodes probabilistic relationships among a set of random variables in relational data. Random variables in $G_M(V_M, E_M)$ are usually descriptive attributes of the dataset and edges between variables is a sign of their correlation. Figure 2 shows an example of a $G_M(V_M, E_M)$ for the instance in Figure 1.

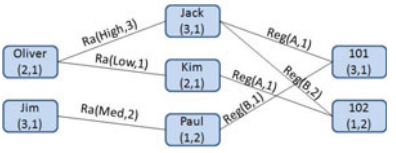

Fig. 1. Graph $G_D(V_D, E_D)$ for an instance of Table 1. Variables represent objects of the schema and links represent the relationships. An array is assigned to nodes and edges that carry the information of the objects and the relationships.

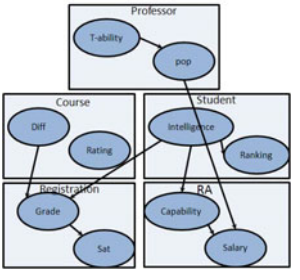

Fig. 2. An example of a graph $G_M(V_M, E_M)$ for G_D given in Figure 1. The variables represent descriptive attributes of tables of the schema and edges show dependencies between attributes.

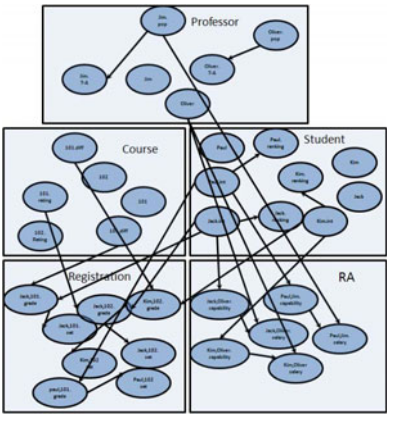

Fig. 3. The graph G_I for G_M in Figure 2 rolled out over the schema in Figure 1. The variables of the graph show descriptive attributes of each object and edges connect attributes of objects using the template from G_M.

3. A third graph G_I used for inference in Relational data mining Models is much larger than the other two graphs. The structure of G_I is determined by G_M and G_D. Let $G_D^{'}(V_D, E_D)$ be the graph of the test data constructed similarly to $G_D(V_D, E_D)$ of the instance and $G_M(V_M, E_M)$ be the model. Then, $G_I(V_I, E_I)$ takes the template from G_M and the relations in $G_D^{'}(V_D, E_D)$ constrains the way that G_I is rolled out. Figure 3 shows G_I for G_D in Figure 1 using the template G_M from Figure 2. The most common query type is the standard conditional probability

query $p(Y = y|E = e)$ where the evidence, E, is a subset of the random variables with instantiation e to those variables and the query nodes.

5 Statistical Relational Learning Models

In this section we review four of state-of-the-art models of statistical relational learning.

5.1 Probabilistic Relational Models (PRM)

Probabilistic relational models (PRMs) [8] are a rich representation language for statistical models which are among the first successful methods proposed for statistical relational learning (SRL). PRMs combine logical representation with probabilistic semantics based on directed graphical models. PRMs extend Bayesian networks with the concept of objects to deal with relational data.

Representation. A PRM, like a Bayesian network, has two components: the dependency structure and the parameters associated with it. For most cases in $G_M(V_M, E_M)$ where random variables correspond to attributes from different tables, an edge between them shows a correlation between sets. For example, ranking of a student may depend on the grades of courses she has taken. PRMs use the notion of aggregation from database theory to summerize information from different links (*mode, mean, median, maximum, and minimum*) We consider two types of PRMs

- PRMs in which both the objects and their relationships are fixed and the only uncertainty is over descriptive attributes of entities and relationships. Such a PRM and a database of objects with their relationships define a probability distribution over the descriptive attributes of the objects.
- PRMs with structural uncertainty in which objects are fixed but there is uncertainty over objects to which relationships correspond to. For example the tables for $student$, $course$, and $registration$ are available, however the foreign key attributes in the registration table which determines the student and the course that the information in the tuple refers to is missing.

Parameter Learning. The key feature in parameter learning is the likelihood function which is defined as the probability of the data given the model. Let $G_M^{\{V\}}(o)$ be an assignment of values to attributes related to object o over a model m. For example, $G_M^{intelligence,ranking}(Jack) = \{3, 1\}$. Formula 2 shows the likelihood of data given the model for PRMs.

$$l(\theta_{G_M}|G_D, G_M) = \sum_{v \in V_M} \sum_{o} \log p(G_M^v(o)|G_M^{pa(v)}(o)) \tag{2}$$

Where θ_{G_M} indicates the parameters for G_M, o is the set objects. In order to do parameter learning, the well understood theory of learning maximum likelihood(ML) parameter estimation may be used. Using ML, formula 2 can be decomposed into summation terms that each may be maximized separately, where $C_{[v,u]}$ is the number of

times $G_M^V(o) = v$ and $G_M^{pa(V)}(o) = u$ occurs, and $v|u$ is the conditional probability of $G_M^V(o) = v$ given $G_M^{pa(V)}(o) = u$.

$$l(\theta_{G_M}|G_D, G_M) = \sum_{v \in V_M} \sum_{k \in D(v)} \sum_{u \in D(pa(v))} C_{[v,u]} \times \log \theta_{G_M(v|u)} \tag{3}$$

Structure Learning. Structure learning is more challenging task than parameter learning in graphical models. The general task of structure learning in PRMs is to find the set of edges E_M in G_M. The "goodness" of different structures must be comparable in order to allow preference of a model over another. For evaluating different structures, maximum a posteriori (MAP) or score functions like BIC [10] are used. For Bayesian networks the task of finding the best structure is NP hard. PRMs use greedy algorithms that iteratively modifies the structure to increase the score. Three operations used in each step are adding, removing or reversing an edge. In each step, all possible transformations using these operations are considered. The structure with highest score is chosen as the next candidate.

Inference. PRMs in few cases, when either the skeleton is small or the tree width is low, can use exact inference on the $G_I(V_I, E_I)$ graph. Unfortunately exact inference is usually not applicable with real world data. Inference in PRMS requires inference over the ground network defined by an instantiated PRM for a specific skeleton. Because $G_I(V_I, E_I)$ is usually very large, efficient inference is very complicated. The approximate algorithm used for inference in PRMs is a variant of belief propagation [19, 26].

5.2 Relational Dependency Network (RDN)

Relational Dependency Networks(RDNs)[20], an extension of Dependency Networks (DNs)[11], are a class of graphical models that approximate a joint distribution using a bidirected graph with conditional probability tables for variables.DNs have several characteristics which make them favorable for relational data: Their unique representation provides the ability to represent cyclic dependencies, simple methods for parameter estimation, and efficient structure learning techniques. The strength of RDNs is mostly due to the use of pseudo-likelihood [1] learning algorithm that estimates an acceptable approximation of the joint distribution.

Representation. RDNs extend the graphical model of DNs to the relational setting. The model encodes probabilistic relationships among a set of random variables with a bidirected graph $G_M(V_M, E_M)$ where conditional independence is interpreted using graph separation as in undirected models. Although conditional independence is inferred using an undirected view of the graph, bidirected edges are used to define the set of neighbors of a node used in their CPT. Each node has a probability distribution conditional on its neighbors as in directed models.The nodes of the model are similar to the variables in PRMs. Each node $v \in V$ corresponds to a descriptive attribute from the entity or relationship tables. An Edge between two nodes correspond to correlations between attributes of the dataset. However, the conditional probability distributions do not factor the joint probability so its impossible to calculate the joint probability directly as in PRMs.

Learning. RDNs extend the learning of DNs to a relational setting. The set of the conditional probability tables CPTs describe both the structure and the parameters of the model. RDNs use pseudo-likelihood techniques [1] to avoid the complexities of estimating the partition function. Instead of optimizing the log-likelihood of the joint distribution, RDNs optimize the pseudo-likelihood for each node separately conditioned on all its neighbors. Formula 4 computes the pseudo-likelihood for each node in G_M seperately, where θ_{G_M} is the parameters for G_M, $D(v)$ is the domain of values for variable v, and $D(p(v))$ is the domain of values for parents of v.

$$Pl(\theta_{G_M}|G_D, G_M) = \sum_{v \in V_M} \sum_{k \in D(v)} \sum_{u \in D(pa(v))} P(v = k|pa(v) = u) \tag{4}$$

where, $P(v = k|pa(v) = u)$is computed by two main relational learners

- Relational Bayesian classifier is a non selective model that treats heterogeneous relational subgraphs as a homogeneous set of attribute multisets. The classifier assumes that each value in the multiset is drawn independently from the same multinomial distribution. For example, the *difficulty* of the courses taken by a student form a multiset {Hard, Hard, Easy, Medium}. RBC selects values independently from the multiset distribution.
- Relational probability trees are a selective model that extend traditional classification trees to relational settings. Relational probability trees also treat heterogeneous relational subgraphs as a set of attribute multisets; however, instead of treating the values like an independent set, Relational probability trees use aggregation functions to map the set of values into a single value.

Inference. RDNs use Gibbs sampling for inference on G_I. The values of unobserved variables are initialized with their prior distribution and are iteratively relabeled using the current state on the model and the CPT of the node. Gibbs sampling is generally an inefficient approach to estimate the joint probability of the model, however, it is reasonably fast to estimate conditional probabilities for each node given its parents.

5.3 Bayesian Logic Programming (BLP)

Bayesian Logic programs [15] are a model based on Bayesian networks. BLPs use logic programming [18] to unify Bayesian networks with logic programming. This unification overcomes the propositional character of Bayesian networks and logical programs. BLPs use Bayesian clauses that use a conditional probability table to present the distribution of the head of the clause conditional on its body, and use combining rules to unite the information on a single literal that is the head of several clauses. BLPs are implemented in a software called BALIOS [15] and are considered as one of successful models of SRL.

Representation. BLPs are produced from logical programs. A logical program is a set of clauses of the form $A : B_1, B_2, \ldots B_n$ where A and B_i are universally quantified atoms. We call A the head and B_is the body of the clause. The head of the clause is considered true in the model if the body of the of the clause is entailed. BLPs use

Bayesian clauses which differ from logical clauses. Bayesian clauses uses a conditional probability table to keep the probability of the head of the clause conditioned on its body, whereas logical clauses have a deterministic value. It is possible to have several clauses with the same variable in the head of the clause. Since each clause has its own local probability distribution, a variable may have several local probability distributions with possibly different sets of parents. To obtain a single conditional probability distribution for the variable that includes the union of all parents, *combining rules* are used. A combining rule is a function that maps finite sets of conditional probability distributions $\{ P(A|A_{i1} \dots A_{in_i}), i = 1 \dots m \}$ on to one combined conditional probability distribution $P(A|B_1 \dots B_k)$ with $P(A|B_1 \dots B_k) \subseteq \bigcup_{i=1}^{m} \{A_{i1} \dots A_{in_i}\}$.

Learning. Learning in BLPs, as in MLNs [4], is a probabilistic extension of learning in inductive logic programing [18] and is formulated as follows. "Given a set of Bayesian logic programs H, G_D, and a scoring function F; find a acyclic candidate $H^* \in H$ such that H^* matches G_D best according to F ". The score function F is used to evaluate how good the clauses are. To adapt traditional techniques used for parameter estimation of Bayesian networks such as Expectation maximization algorithm, combining rules are required to be decomposable ; most common combining rules for Bayesian networks such as "noisy or" are decomposable. The best match refers to those parameters of the associated conditional probability distributions that maximize the scoring function where the score function is based on maximum likelihood [5]. Structure learning in BLNs follows the procedure of rule learning in ILP systems [21] which have operators such as adding and deleting logically valid literals, flipping, instantiating variables, unifying variables on literals or clauses. BLNs speed up the learning procedure executing several operations simultaneously.

Inference. Inference, as in other SRL methods, is intractable in BLNs and is proceeded via grounding the clauses of Bayesian logic Program. Each Bayesian logic Program species a propositional Bayes net, where inference is done using standard Bayes net learning algorithms

5.4 Markov Logic Networks (MLN)

Markov Logic Networks (MLNs) [4] are among the most well known methods proposed for statistical relational learning. Syntactically MLNs extend first-order logic and put a weight for each formula. Semantically, they can represent a probability distribution over possible worlds using formulas and their coressponding weights.

Representation. Formally, a Markov Logic Network is a set of pairs of formulas and their corresponding weights (F_i, w_i) where formulas are in first order logic and the weights are any real number. The set of formulas in MLNs correspond to the class model G_M. An MLN with a finite set of objects in G_D defines a ground Markov network G_I. A grounding is defined as assigning a value to a variable from its domain. G_I has a binary node for each ground predicate in the MLN. The value of the node is 1 if the ground atom is true and 0 otherwise. Also, there is an edge between two nodes if the ground predicates appear together in at least one grounding of a formula. An MLN is a template for the ground Markov network and the size of the model is a function of

the number of objects. The ground network has regularities in structure and parameters which are forced by the MLN. All terms in a formula form a clique.

A world is an assignment of truth values to all possible ground atoms. Each state of the Markov network presents a possible world. The probability distribution over possible worlds x specified by the ground network is calculated by Formula 5 where $n_i(x)$ is the number of true groundings for F_i in x and Z is the partition function that is used to make the summation of all possible groundings adds up to one.

$$P(X = x) = \frac{1}{Z} \exp(\sum_i w_i n_i(x)) \tag{5}$$

Parameter Learning. Finding the weight of the formulas in MLNs is equivalent to computing parameters in other models. The weights are learned from the relational database. Assuming the network has n ground atoms, a database has up to n entries indicating the true facts. MLNs use the close-world assumption [6] that if a ground atom is absent in the database, it is assumed to be false. In MLNs, the weight are computed by maximizing the log likelihood of the data. Formula uses the derivative of Formula 5 with respect to its weights.

$$\frac{\partial}{\partial w_i} \log p_w(x) = n_i(x) - \sum_{x'} P_w(x') n_i(x') \tag{6}$$

Where the sum is over all possible databases x', and $P_w(X = x')$ is $P(X = x')$ computed using the current weight vector. However, Formula 6 is NP hard to compute. An approximation for calculating probability of a world using pseudo-likelihood [1] is used that omits the use of the partition function. Pseudo-likelihood is a measure in statistics that serves as an approximation of the distribution of x based on its Markov blanket instead of all other x'.

Structure Learning. Structure learning in MLNs is very similar to BLPs. They use the CLAUDIEN [3] system which is able to learn first order formulas and not just horn clauses.

Inference. Inference has two main phases in MLNs. In the first phase, a minimal subset of G_I the ground Markov network is selected. Many predicates that are independent of the predicates of the query may be filtered in this phase. As a result inference carried out over a smaller Markov network. In the second phase inference is performed on the Markov network using Gibbs sampling [2] where the evidence nodes are observed and are set to their values. Gibbs sampling first randomly initialize and orders unobserved variables in the network, i.e. $\{X_1 = x_1 \dots X_n = x_n\}$, and then iterate through the variables. In step 1 a new value x'_1 for variable X_1 is sampled conditional on all the other variables $P(X_1|X_2 = x_2 \dots X_n = x_n)$, and in step i a new value x'_i for variable x_i is sampled conditional on all the other variables $P(X_i|X_1 = x'_1 \dots X_{i-1} = x'_{i-1}, X_{i+1} = x_{i+1} \dots X_n = x_n)$. Conditional independence eases the computation as V_i conditional on its immediate neighbors, Markov Blanket, is independent of all other variables.

6 Limitations of Current Methods and Conclusion

Statistical relational models(SRL) methods have generally been very successful. Table 2 summarizes comparison among different SRL models based on various dimensions of importance in Statistical-Relational Learning. Many different problems have been defined over SRL models and good results have already been achieved; however, we believe limitations on current models show the necessity for more research on the field. In this section we point out some of the limitations of most SRL models.

Table 2. A comparison of Probabilistic Relational Models (PRMs), Markov Logic Networks (MLNs), Relational Dependency Netoworks (RDNs), and Bayes Logic Networks (BLNs) along various dimensions of importance in Statistical-Relational Learning

	PRM	**MLN**	**RDN**	**BLN**
Class level model	Directed GM	Logical clauses	Bidirected GM	bipartite directed GM
Parameter Estimation	ML to fill CPTs	ML to learn Weights	CR Learners to learn CPTs	ML to fill CPTs
Structure learning	Score Based learning	ILP methods	Use CR learners	ILP Methods
Inference Graph	Bayesian network	Markov Model	undirected model	Bayesian network
Inference Method	belief propagation	Pseudo-likelihood	Pseudo-likelihood	stand BN inference
Autocorrelation	self-loops in class-level model	additional variables needed	Yes	Not discussed
X-many relationships	Require Aggregation	No requirements	No requirements	Require combination rules

The computational complexity of inference is probably the biggest limitation shared between most SRL methods. The size of the graph G_I is proportional to the number of descriptive attributes and objects, which limits the scalability for many realistic datasets. PRMs and BLNs suffer from inference as they use standard complex Bayesian network inference algorithms on the G_I graphs. MLNs share the same problem as they use an undirected model as their ground network. It is impractical to do exact inference on large Markov models because of the computations on the partition function. MLNs are forced into using approximation techniques for inference on generative models. Pseudo-likelihood fails to give significant results when querying on variables that are distant in the model [4]. Inference is quicker in RDNS, because they approximate a joint distribution using a bi-directed graph with conditional probability tables for variables; however, the conditional probability distributions do not factor the joint probability so it is impossible to calculate the joint probability directly as in most other SRL methods. A lot of research is currently being done on lifted inference [16]. Lifted inference aims to do exact inference without materializing the ground inference graph.

Autocorrelation causes significant representation and computational difficulties for most SRL models. PRMs and BLNs add a single random variable to their class model for every descriptive attribute in the dataset. Due to this fact, autocorrelation is shown with self loops in their class model. It is possible to achieve acyclic ground models even with the existence of self loops in the class model, however additional information on the dataset is required which complicates the model and is usually unknown or does not exist. A well known example is the correlation between blood type of parents and their children. The parent relationship does not introduce loops so, showing the correlation does not have any cycles in the ground model, however, the descriptive attribute of blood type requires a self loop in the class level model. Neville and Jensen conclude

that the acyclicity constraints of directed PRMs precludes the learning of arbitrary autocorrelation dependencies and thus severely limits the applicability of these models in relational domains[20].

The underlying complicated multi-table structure results in large datasets, which leads to difficulties of scalability and efficiency in structure learning of SRL. Structure learning in MLNs is very similar to ILP methods which are usually not scalable and very inefficient for large datasets. Some new approaches for structure learning in MLNs are being considered, but they have only been tested on small to medium sized datasets. PRMS and BLNs use score based learning that leads to local maxima solutions. RDNs use two relational learners as part of their model and both relational learners have its own problems. Relational Bayesian classifiers are non selective and choose a single value independently from the multinomial distribution of values of the link to be a representer, but this weakens the model. Relational probability trees use an informative aggregation functions to describe a set of values with one value, but this increases the complexity of the learning procedure.

Acknowledgments. This article is the result of a part of our research under the supervision of Dr Oliver Schulte. We thanks Dr Schulte for his guidance and interesting discussions. We also thank Tong Man and Xiaoyuan Xu for helpful comments.

References

[1] Besag, J.: Statistical analysis of non-lattice data. The Statistician 24(3), 179–195 (1975)
[2] Casella, G., George, E.I.: Explaining the gibbs sampler. The American Statistician 46(3), 167–174 (1992)
[3] De Raedt, L., Dehaspe, L.: Clausal discovery. Mach. Learn. 26(2-3), 99–146 (1997)
[4] Domingos, P., Richardson, M.: Markov logic: A unifying framework for statistical relational learning. In: Introduction to Statistical Relational Learning [9], ch. 12, pp. 339–367 (2007)
[5] Edgeworth, F.Y.: On the probable errors of frequency-constants (contd.). Journal of the Royal Statistical Society 71(3), 499–512 (1908)
[6] Eiter, T., Gottlob, G.: Propositional circumscription and extended closed world reasoning are π_2^p-complete. Theoretical Computer Science 114, 231–245 (1993)
[7] Flake, G.W., Lawrence, S., Lee Giles, C.: Efficient identification of web communities. In: KDD 2000: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 150–160. ACM, New York (2000)
[8] Getoor, L., Friedman, N., Koller, D., Pfeffer, A., Taskar, B.: Probabilistic relational models. In: Introduction to Statistical Relational Learning [9]
[9] Getoor, L., Tasker, B.: Introduction to statistical relational learning. MIT Press, Cambridge (2007)
[10] Heckerman, D.: A tutorial on learning with bayesian networks. In: NATO ASI on Learning in graphical models, pp. 301–354 (1998)
[11] Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R., Kadie, C., Kaelbling, P.: Dependency networks for inference, collaborative filtering, and data visualization. Journal of Machine Learning Research 1, 49–75 (2000)
[12] Jensen, D., Neville, J.: Linkage and autocorrelation cause feature selection bias in relational learning (2002). In: Proceedings of the 19th International Conference on Machine Learning (2002)

[13] Jensen, D., Neville, J., Gallagher, B.: Why collective inference improves relational classification. In: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 593–598 (2004)
[14] Jordan, M.: Graphical models. Statistical Science (Special Issue on Bayesian Statistics) 19, 140–155 (2004)
[15] Kersting, K., de Raedt, L.: Bayesian logic programming: Theory and tool. In: Introduction to Statistical Relational Learning [9]
[16] Zettlemoyer, L.S., Leslie, M.H., Kristian, P.K., Kersting, B.M.: Reasoning about large populations with lifted probabilistic inference. In: NIPS Workshop (2008)
[17] Lauritzen, S.L.: Graphical Models. Oxford Statistical Science Series. Oxford University Press, USA (July 1996)
[18] Muggleton, S.: Inductive logic programming. New Gen. Comput. 8(4), 295–318 (1991)
[19] Murphy, K.P., Weiss, Y., Jordan, M.I.: Loopy belief propagation for approximate inference: An empirical study. In: Proceedings of Uncertainty in AI, pp. 467–475 (1999)
[20] Nevile, J., Jensen, D.: Relational dependency networks. In: An Introduction to Statistical Relational Learning [9]
[21] Nienhuys-Cheng, S.-H., de Wolf, R. (eds.): Foundations of Inductive Logic Programming. LNCS, vol. 1228. Springer, Heidelberg (1997)
[22] Pearl, J.: Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann, San Francisco (1988)
[23] Taskar, B., Wong, M., Abbeel, P., Koller, D.: Link prediction in relational data (2004)
[24] Ullman, J.D.: Principles of database systems, Vol. 2. Computer Science Press, Rockville (1982)
[25] Wang, Y., Wang, Y., Kitsuregawa, M.: Link based clustering of web search results. LNCS, pp. 225–236. Springer, Heidelberg (2001)
[26] Weiss, Y.: Correctness of local probability propagation in graphical models with loops. Neural Comput. 12(1), 1–41 (2000)
[27] Winkler, W.: The state of record linkage and current research problems (1999)