Statistical Relational Learning

Luc De Raedt[1] and Kristian Kersting[2]

[1] K.U. Leuven, Leuven, Beligum
[2] Fraunhofer IAIS, Sankt Augustin, Germany

Related keywords: Bayesian Networks, Markov Networks, Relational Learning, Inductive Logic Programming, Statistical Learning

Definition

Statistical relational learning aka. probabilistic inductive logic programming deals with machine learning and data mining in relational domains where observations may be missing, partially observed, or noisy. In doing so, it addresses one of the central questions of artificial intelligence – the integration of probabilistic reasoning with machine learning and first order and relational representations – and deals with all related aspects such as reasoning, parameter esimtation, and structure learning.

Motivation and Background

One of the central questions of artificial intelligence is concerned with combining expressive knowledge representation formalisms such as relational and first-order logic with principled probabilistic and statistical approaches to inference and learning. While traditionally relational and logical representations, probabilistic and statistical reasoning, and machine learning have been studied independently of one another, statistical relational learning investigates them jointly, cf. Figure 1. A major driving force is the explosive growth in the amount of heterogeneous data that is being collected in the business and scientific world in domains such as bioinformatics, transportation systems, communication networks, social network analysis, citation analysis, and robotics. Characteristic for these domains is that they provide *uncertain* information about varying numbers of entities and relationships among the entities, that is, about *relational* domains. Traditional machine learning approaches are able to cope either with uncertainty or with relational representations but typically not with both.

Many formalisms and representations have been developed in statistical relational learning. For instance, Eisele [1] has introduced a probabilistic variant of Comprehensive Unification Formalism (CUF). In a similar manner, Muggleton [2] and Cussens [3] have upgraded stochastic grammars towards *stochastic logic programs*. Sato [4] has introduced *probabilistic distributional semantics* for logic programs. Taskar *et al.* [5] have upgraded Markov networks towards *relational Markov networks*, and Domingos and Richardson [6] towards *Markov logic networks*. Neville and Jensen [7] have extended dependency networks towards *relational dependency networks*. Another research stream has investigated

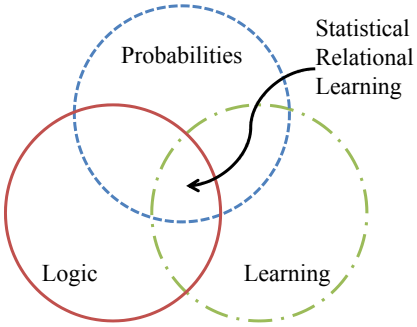

Fig. 1. Statistical Relational Learning aka. probabilistic inductive logic programming combines probability, logic, and learning.

logical and relational extensions of Bayesian networks. It includes Poole's *independent choice Logic* [8], Ngo and Haddawy's *probabilistic logic programs* [9], Jäger's *relational Bayesian networks* [10], Koller, Getoor, and Pfeffer's *probabilistic relational models* [11, 12], and Kersting and De Raedt's *Bayesian logic programs* [13].

The benefits of employing logical abstraction and relations within statistical learning are manyfold:

- Relations among entities allow one to use information about one entity to help reach conclusions about other, related entities.
- Variables, that is, placeholders for entities allow one to make abstraction of specific entities.
- Unification allows one to share information among entities. Thus, instead of learning regularities for each single entity independently, statistical relational learning aims at finding general regularities among groups of entities.
- The learned knowledge is often declarative and compact, which makes it easier for people to understand and to validate.
- In many applications, there is a rich background theory available, which can efficiently and elegantly be represented as a set of general regularities. This is important because background knowledge may improve the quality of learning as it focuses the learning on the relevant patterns, that is, it restricts the search space.
- When learning a model from data, relational and logical abstraction allow one to reuse experience in that *learning about one entity improves the prediction for other entities*; and this may even generalize to objects that have never been observed before.

Thus, relational and logical abstraction make statistical learning more robust and efficient. This has proven to be beneficial in many fascinating real-world ap-

plications in citation analysis, web mining, natural language processing, robotics, bio- and chemo-informatics, electronic games, and activity recognition.

Theory

Whereas most of the existing works on statistical relational learning have started from a statistical and probabilistic learning perspective and extended probabilistic formalisms with relational aspects, statistical relational learning can elegantly be introduced by starting from inductive logic programming [14, 15], which is often also called *multi-relational data mining* (MRDM) [16]. Inductive logic programming is a research field at the intersection of machine learning and logic programming. It forms a a formal framework and has introduced practical algorithms for inductively learning relational descriptions (in the form of logic programs) from examples and background knowledge. So, the only difference to statistical relational learning is that it does not explicitly deal with uncertainty.

Essentially, there only two changes to apply to inductive logic programming approaches in order to arrive at statistical relational learning:

1. clauses (that is, logical formulae that can be interpreted as rules; cf. below) are annotated with probabilistic information such as conditional probabilities, and
2. the covers relation (which states the conditions under which a hypothesis considers an example as positive) becomes probabilistic.

A probabilistic covers relation softens the hard covers relation employed in traditional inductive logic programming and is defined as the probability of an example given the hypothesis and the background theory.

Definition 1 (Probabilistic Covers Relation). *A probabilistic covers relation takes as arguments an example e, a hypothesis H and possibly the background theory B, and returns the probability value $\mathbf{P}(e \mid H, B)$ between 0 and 1 of the example e given H and B, i.e.,* $\text{covers}(e, H, B) = \mathbf{P}(e \mid H, B)$.

It specifies the likelihood of the example given the hypothesis and the background theory. Different choices of the probabilistic covers relation lead to different statistical relational learning approaches; this is akin to the learning settings in inductive logic programming.

Statistical Relational Languages

There is a multitude of different languages and formalisms for statistical relational learning. For an overview of these languages we refer to [17, 18]. Here, we choose two formalisms that are representatives of the two main streams in statistical relational learning. First, we discuss Markov logic [6], which upgrades Markov network towards first order logic, and secondly, we discuss ProbLog [19], which is a probabilistic Prolog based on Sato's distribution semantics [4]. While Markov logic is a typical example of knowledge based model construction, ProbLog is a probabilistic programming language.

Case Study: Markov Logic Networks Markov logic combines first-order logic with Markov networks. The idea is to view logical formulae as soft constraints on the set of possible worlds, that is, on the interpretations (an interpretation is a set of facts). If an interpretation does not satisfy a logical formula, it becomes less probable, but not necessarily impossible as in traditional logic. Hence, the more formulae an interpretation satisfies, the more likely it becomes. In a Markov logic network, this is realized by associating a weight to each formula that reflects how strong the constraint is. More precisely, a Markov logic network consists of a set of weighted clauses[3] $H = \{c_1, \ldots, c_m\}$. The weights w_i of the clauses then specify the strength of the clausal constraint.

Example 1. Consider the following example (adopted from [6]). Friends & Smokers is a small Markov logic network that computes the probability of a person having lung cancer on the basis of her friends smoking. This can be encoded using the following weighted clauses:

$1.5 : \mathsf{cancer(P)} \leftarrow \mathsf{smoking(P)}$
$1.1 : \mathsf{smoking(X)} \leftarrow \mathsf{friends(X,Y)}, \mathsf{smoking(Y)}$
$1.1 : \mathsf{smoking(Y)} \leftarrow \mathsf{friends(X,Y)}, \mathsf{smoking(X)}$

The first clause states the soft constraint that smoking causes cancer. So, interpretations in which persons that smoke have cancer are more likely than those where they do not (under the assumptions that other properties remain constant). The second and third clauses state that friends of smokers are typically also smokers.

A Markov logic network together with a Herbrand domain (in the form of a set of constants $\{d_1, \ldots, d_k\}$) then induces a grounded Markov network, which defines a probability distribution over the possible Herbrand interpretations.

The nodes, that is, the random variables in the grounded network, are the atoms in the Herbrand base, that is, the facts of the form $p(d'_1, \cdots, d'_n)$ where p is a predicate or relation and the d'_i are constants. Furthermore, for every ground instance $c_i\theta$ of a clause c_i in H, there will be an edge between any pair of atoms $a\theta, b\theta$ that occurs in $c_i\theta$. The Markov network obtained for the constants anna and bob is shown in Fig. 2. To obtain a probability distribution over the Herbrand intepretations, we still need to define the potentials. The probability distribution over intepretations I is

$$\mathbf{P}(I) = \frac{1}{Z} \prod_{c:clause} \mathbf{f}_c(I) \tag{1}$$

where the f_c are defined as

$$f_c(I) = e^{n_c(I) w_c} \tag{2}$$

[3] Markov logic networks, in principle, also allow one to use arbitrary logical formulae, not just clauses. However, for reasons of simplicity, we only employ clauses and make some further simplifications.

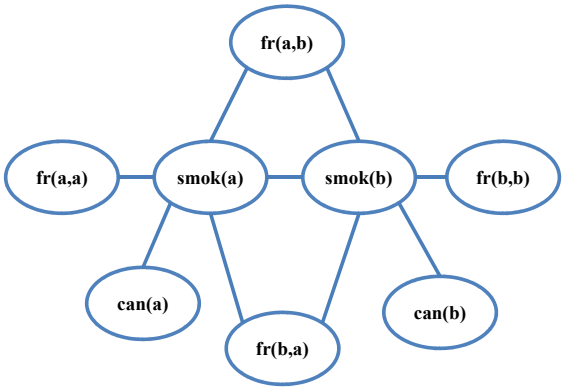

Fig. 2. The Markov network for the constants ann and bob. Adapted from [6].

and $n_c(I)$ denotes the number of substitutions θ for which $c\theta$ is satisfied by I, and Z is a normalization constant. The definition of a potential as an exponential function of a weighted feature of a clique is common in Markov networks; cf. graphical models. The reason is that the resulting probability distribution is easier to manipulate.

Note that for different (Herbrand) domains, different Markov networks will be produced. Therefore, one can view Markov logic networks as a kind of template for generating Markov networks, and, hence, Markov logic is based on knowledge-based model construction. Notice also that Markov logic networks define a probability distribution over interpretations, and nicely separate the qualitative from the quantitive component.

Case Study: ProbLog Many formalisms do not explicitly encode a set of conditional independency assumptions, as in Bayesian or Markov networks, but rather extend a (logic) programming language with probabilistic choices. Stochastic logic programs [2, 20] directly upgrade stochastic context-free grammars towards definite clause logic, whereas PRISM [4], Probabilistic Horn Abduction [8] and the more recent Independent Choice Logic (ICL) [21] specify probabilities on facts from which further knowledge can be deduced. As a simple representative of this stream of work, we introduce the probabilistic Prolog called ProbLog [19].

The key idea underlying Problog is that some facts f for *probabilistic* predicates are annotated with a probability value. This value indicates the degree of belief, that is the probability, that any ground instance $f\theta$ of f is true. It is also assumed that the $f\theta$ are marginally independent. The probabilistic facts are then augmented with a set of definite clauses defining further predicates (which

should be disjoint from the probabilistic ones). An example adapted from [19] is given below.

Example 2. Consider the facts

$0.9 : \mathsf{edge}(\mathsf{a},\mathsf{c}) \leftarrow$
$0.7 : \mathsf{edge}(\mathsf{c},\mathsf{b}) \leftarrow$
$0.6 : \mathsf{edge}(\mathsf{d},\mathsf{c}) \leftarrow$
$0.9 : \mathsf{edge}(\mathsf{d},\mathsf{b}) \leftarrow$

which specify that with probability 0.9 there is an edge from a to c. Consider also the following (simplified) definition of path/2.
$\mathsf{path}(\mathsf{X},\mathsf{Y}) \leftarrow \mathsf{edge}(\mathsf{X},\mathsf{Y})$
$\mathsf{path}(\mathsf{X},\mathsf{Y}) \leftarrow \mathsf{edge}(\mathsf{X},\mathsf{Z}),\mathsf{path}(\mathsf{Z},\mathsf{Y})$

One can now define a probability distribution on (ground) proofs as follows. The probability of a ground proof is the product of the probabilities of the (ground) clauses (here, facts) used in the proof. For instance, the only proof for the goal $\leftarrow \mathsf{path}(\mathsf{a},\mathsf{b})$ employs the facts $\mathsf{edge}(\mathsf{a},\mathsf{c})$ and $\mathsf{edge}(\mathsf{c},\mathsf{b})$; these facts are marginally independent, and hence the probability of the proof is $0.9 \cdot 0.7$. The probabilistic facts used in a single proof are sometimes called an *explanation*.

It is now tempting to define the probability of a ground atom as the sum of the probabilities of the proofs for that atom. However, this does not work without additional restrictions, as shown in the following example.

Example 3. The fact $\mathsf{path}(\mathsf{d},\mathsf{b})$ has two explanations:

- $\{\mathsf{edge}(\mathsf{d},\mathsf{c}), \mathsf{edge}(\mathsf{c},\mathsf{b})\}$ with probability $0.6 \times 0.7 = 0.42$, and
- $\{\mathsf{edge}(\mathsf{d},\mathsf{b})\}$ with probability 0.9.

Summing the probabilities of these explanations gives a value of 1.32, which is clearly impossible.

The reason for this problem is that the different explanations are not mutually exclusive, and therefore their probabilities may not be summed. The probability $P(\mathsf{path}(\mathsf{d},\mathsf{b}) = true)$ is, however, equal to the probability that *a* proof succeeds, that is

$$P(\mathsf{path}(\mathsf{d},\mathsf{b}) = true) = P[(\mathsf{e}(\mathsf{d},\mathsf{c}) \wedge \mathsf{e}(\mathsf{c},\mathsf{b})) \vee \mathsf{e}(\mathsf{d},\mathsf{b})]$$

which shows that computing the probability of a derived ground fact reduces to computing the probability of a boolean formula in disjunctive normal form (DNF), where all random variables are marginally independent of one another. Computing the probability of such formulae is an NP-hard problem, the *disjoint-sum* problem. Using the *inclusion-exclusion* principle from set theory, one can

compute the probability as

$$\begin{aligned} P(\mathsf{path}(\mathsf{d},\mathsf{b}) = \mathit{true}) &= P[(\mathsf{e}(\mathsf{d},\mathsf{c}) \wedge \mathsf{e}(\mathsf{c},\mathsf{b})) \vee \mathsf{e}(\mathsf{d},\mathsf{b})] \\ &= P(\mathsf{e}(\mathsf{d},\mathsf{c}) \wedge \mathsf{e}(\mathsf{c},\mathsf{b})) + P(\mathsf{e}(\mathsf{d},\mathsf{b})) \\ &\quad -P((\mathsf{e}(\mathsf{d},\mathsf{c}) \wedge \mathsf{e}(\mathsf{c},\mathsf{b})) \wedge \mathsf{e}(\mathsf{d},\mathsf{b})) \\ &= 0.6 \times 0.7 + 0.9 - 0.6 \times 0.7 \times 0.9 = 0.942 \end{aligned}$$

There exist more effective ways to compute the probability of such DNF formulae; cf. [19], where binary decision diagrams are employed to represent the DNF formula.

The above example shows how the probability of a specific fact is defined and can be computed. The distribution at the level of individual facts (or goals) can easily be generalized to a possible world semantics, specifying a probability distribution on interpretations. It is formalized in the *distribution semantics* of [4], which is defined by starting from the set of all probabilistic ground facts F for the given program. For simplicity, we shall assume that this set is finite, though Sato's results also hold for the infinite case. The distribution semantics then starts from a probability distribution $P_F(S)$ defined on subsets $S \subseteq F$:

$$P_F(S) = \prod_{f \in S} P(f) \prod_{f \notin S} (1 - P(f)) \tag{3}$$

Each subset S is now interpreted as a set of logical facts and combined with the definite clause program R that specifies the logical part of the probabilistic logic program. Any such combination $S \cup R$ possesses a unique least Herbrand model $M(C)$, which corresponds to a possible world. The probability of such a possible world is then the sum of the probabilities of the subsets S yielding that possible world, that is:

$$P_W(M) = \sum_{S \subseteq F : M(S \cup R) = M} P_F(S) \tag{4}$$

For instance, in the path example, there are 16 possible worlds, which can be obtained from the 16 different truth assignments to the facts, and whose probabilities can be computed using Equation (4). As for graphical models, the probability of any logical formulae can be computed from a possible world semantics (specified here by P_W).

Because computing the probability of a fact or goal under the distribution semantics is hard, systems such as PRISM [4] and Probabilistic Horn Abduction (PHA) [8] impose additional restrictions that can be used to improve the efficiency of the inference procedure. The key assumption is that the explanations for a goal are *mutually exclusive*, which overcomes the disjoint-sum problem. If the different explanations of a fact do not overlap, then its probability is simply the sum of the probabilities of its explanations. This directly follows from the inclusion-exclusion formulae as under the exclusive-explanation assumption the conjunctions (or intersections) are empty

Learning

Essentially, any statistical relational approach can be viewed as lifting a traditional inductive logic programming setting by associating probabilistic information to clauses and by replacing the deterministic coverage relation by a probabilistic one. In contrast to traditional graphical models such as Bayesian networks or Markov networks, however, we can also employ "counterexamples" for learning. Consider a simple kinship domain. Assume `rex` is a male person. Consequently, he cannot be the `daughter` of any other person, say `ann`. Thus, `daughter(rex, ann)` can be listed as a negative example although we will never observe it. "Counterexamples" conflict with the usual view on learning examples in statistical learning.

In statistical learning, we seek to find that hypothesis H^*, which is most likely given the learning examples:

$$H^* = \arg\max_H P(H|E) = \arg\max_H \frac{P(E|H) \cdot P(F)}{P(E)} \quad \text{with} \quad P(E) > 0 \; .$$

Thus, examples E in traditional statistical learning are always observable, that is, $P(E) > 0$. However, in statistical relational learning, as in inductive logic programming, we may also employ "counterexamples" such as `daughter(rex, ann)`, which have probability "0", and that actually never can be observed.

Definition 2 (SRL Problem). Given *a set $E = E_p \cup E_i$ of positive and negative examples E_p and E_i (with $E_p \cap E_i = \emptyset$) over some example language $\mathcal{L}_E$, a probabilistic covers relation* $\text{covers}(e, H, B) = P(e \mid H, B)$*, a probabilistic logical language $\mathcal{L}_H$ for hypotheses, and a background theory B,* **find** *a hypothesis H^* in $\mathcal{L}_H$ such that $H^* = \arg\max_H \text{score}(E, H, B)$ and the following constraints hold: $\forall\ e_p \in E_p \ : \ \text{covers}(e_p, H^*, B) > 0$ and $\forall\ e_i \in E_i \ : \ \text{covers}(e_i, H^*, B) = 0$. The score is some objective function, usually involving the probabilistic covers relation of the observed examples such as the observed likelihood $\prod_{e_p \in E_p} \text{covers}(e_p, H^*, B)$ or some penalized variant thereof.*

This learning setting unifies inductive logic programming and statistical learning in the following sense: using a deterministic covers relation (either 1 or 0), it yields the classical inductive logic programming learning problem; sticking to propositional logic and learning from *positive* examples, that is, $P(E) > 0$, only yields traditional statistical learning.

To come up with algorithms solving the SRL problem, say for density estimation, one typically distinguishes two subtasks because $H = (L, \lambda)$ is essentially a logical theory L annotated with probabilistic parameters λ:

1. *Parameter estimation* where it is assumed that the underlying logic program L is fixed, and the learning task consists of estimating the parameters λ that maximize the likelihood.
2. *Structure learning* where both L and λ have to be learned from the data.

Below, we will sketch basic parameter estimation and structure learning techniques, and illustrate them for each setting.

Parameter Estimation The problem of parameter estimation is concerned with estimating the values of the parameters λ of a fixed probabilistic program $H = (L, \lambda)$ that best explains the examples E. So, λ is a set of parameters and can be represented as a vector. As already indicated above, to measure the extent to which a model fits the data, one usually employs the likelihood of the data, i.e, $P(E \mid L, \lambda)$, though other scores or variants could be used as well.

When all examples are fully observable, maximum likelihood reduces to frequency counting. In the presence of missing data, however, the maximum likelihood estimate typically cannot be written in closed form. It is a numerical optimization problem, and all known algorithms involve nonlinear optimization The most commonly adapted technique for probabilistic logic learning is the Expectation-Maximization (EM) algorithm [22, 23]. EM is based on the observation that learning would be easy (i.e., correspond to frequency counting), if the values of all the random variables would be known. Therefore, it estimates these values, maximizes the likelihood based on the estimates, and then iterates. More specifically, EM assumes that the parameters have been initialized (e.g., at random) and then iteratively performs the following two steps until convergence:

(E-Step) On the basis of the observed data and the present parameters of the model, it computes a distribution over all possible completions of each partially observed data case.

(M-Step) Treating each completion as a fully observed data case weighted by its probability, it computes the improved parameter values using (weighted) frequency counting.

The frequencies over the completions are called the *expected counts*. Examples for parameter estimation of probabilistic relational model can be found in [17, 18].

Structure Learning The problem is now to learn both the structure L and the parameters λ of the probabilistic program $H = (L, \lambda)$ from data. Often, further information is given as well. As in inductive logic programming, the additional knowledge can take various different forms, including a *language bias* that imposes restrictions on the syntax of L, and an *initial hypothesis* (L, λ) from which the learning process can start.

Nearly all (score-based) approaches to structure learning perform a heuristic search through the space of possible hypotheses. Typically, hill-climbing or beam-search is applied until the hypothesis satisfies the logical constraints and the score(H, E) is no longer improving. The steps in the search-space are typically made using refinement operators, which make small, syntactic modification to the (underlying) logic program.

At this point, it is interesting to observe that the logical constraints often require that the positive examples are covered in the logical sense. For instance, when learning ProbLog programs from entailment, the observed example clauses must be entailed by the logic program.Thus, for a probabilistic program $H = (L_H, \lambda_H)$ and a background theory $B = (L_B, \lambda_B)$ it holds that

$\forall e_p \in E_p$: $P(e|H,B) > 0$ if and only if covers$(e, L_H, L_B) = 1$, where L_H (respectively L_B) is the underlying logic program (logical background theory) and covers(e, L_H, L_B) is the purely logical *covers* relation, which is either 0 or 1.

Applications

Applications of statistical relational learning can be found in many areas such as web search and mining, text mining, bioinformatics, natural language processing, robotics, and social network analysis, among other. Due to space restrictions, we will only name few of these exciting applictions.

For instance, Getoor *et al.* have used statistical relational models to estimate the result size of complex database queries [24]. Segal *et al.* have employed probabilistic relational models for clustering gene expression data [25] and to discover cellular processes from gene expression data [26]. Getoor *et al.* have used probabilistic relational models to understand tuberculosis epidemiology [27]. McGovern *et al.* [28] have estimated probabilistic relational trees to discover publication patterns in high-energy physics. Probabilistic relational trees have also been used to learn to rank brokers with respect to the probability that they would commit a serious violation of securities regulations in the near future [29]. Anguelov *et al.*[30] have used relational Markov networks for segmentation of 3D scan data. They have also been used to compactly represent object maps [31] and to estimate trajectories of people [31]. Kersting *et al.* have employed relational hidden Markov models for protein fold recognition [32]. Poon and Domingos [33] have shown how to use Markov logic to perform joint unsupervised coreference resolution. Xu *et al.* have used non-parametric relational models for analysing social networks [34]. Kersting and Xu have used relational Gaussian processes for learning to rank search results [35]. Recently, Poon and Domingos have shown how to perform unsupervised semantic parsing using Markov logic networks [36].

Current and Future Directions

We have provided an overview of the new and exciting area of statistical relational learning. It combines principles of probabilistic reasoning, logical representation and statistical learning into a coherent whole. The techniques of probabilistic logic learning were analyzed starting from an inductive logic programming perspective by lifting the coverage relation to a probabilistic one and annotating the logical formulae. Different choices of the probabilistic coverage relation lead to different representational formalisms, two of which were introduced.

Statistical relational learning is an active area of research within the machine learning and the artificial intelligence community. First, there is the issue of *efficient inference* and learning. Most current inference algorithms for statistical relational models require explicit state enumeration, which is often impractical: the number of states grows very quickly with the number of domain objects and relations. *Lifted* inference algorithms seek to avoid explicit state enumeration and

directly work at the level of groups of atoms, eliminating all the instantiations of a set of atoms in a single step, in some cases independently of the number of these instantiations. Despite various approaches to lifted inference [37–44], it largely remains a challenging problem. For what concerns learning, advanced principles of both statistical learning and logical and relational learning can be employed for learning the parameters and the structure of probabilistic logics such as statistical *predicate invention* [45] and **boosting** [46]. Recently, people started to investigate *learning from weighted examples*, see e.g. [47] and to link statistical relational learning to support vector machines, see e.g. [48]. Second, there is the issue of **closed-world versus open-world** assumption, i.e., do we know how many objects there are, see e.g. [49] Third, there is interest in dealing with *continuous values* within statistical relational learning, see e.g. [50–53]. This is mainly motivated by the fact that most real-world application actually contain continuous values. *Non-parametric Bayesian* approaches to statistical relational learning have also been developed, see e.g. [54–57], to overcome the typically strong parametric assumptions underlying current statistical relational learning. People have also started to investigate **relational variants of classical statistical learning tasks** such as matrix factorizations, see e.g. [58]. Finally, while statistical relational learning approaches have been used successfully in a number of applications, they do not yet cope with the **dynamic environments** in an effective way.

Recommended Readings

In addition to the references embedded in the text above, we also recommend [17, 18, 15] and the SRL tutorials at major artificial intelligenceI and machine learning conferences.

1. Eisele, A.: Towards Probabilistic Extensions of Contraint-based Grammars. In Dörne, J., ed.: Computational Aspects of Constraint-Based Linguistics Decription-II. DYNA-2 deliverable R1.2.B (1994)
2. Muggleton, S.: Stochastic logic programs. In De Raedt, L., ed.: Advances in Inductive Logic Programming. IOS Press (1996) 254–264
3. Cussens, J.: Loglinear models for first-order probabilistic reasoning. In Blackmond Laskey, K., Prade, H., eds.: Proceedings of the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-99), Stockholm, Sweden, Morgan Kaufmann (1999) 126–133
4. Sato, T.: A Statistical Learning Method for Logic Programs with Distribution Semantics. In Sterling, L., ed.: Proceedings of the Twelfth International Conference on Logic Programming (ICLP-95), Tokyo, Japan, MIT Press (1995) 715 – 729
5. Taskar, B., Abbeel, P., Koller, D.: Discriminative Probabilistic Models for Relational Data. In Darwiche, A., Friedman, N., eds.: Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI-02), Edmonton, Alberta, Canada (August 1-4 2002) 485–492
6. Richardson, M., Domingos, P.: Markov logic networks. Machine Learning **62** (2006) 107–136

7. Neville, J., Jensen, D.: Dependency Networks for Relational Data. In Rastogi, R., Morik, K., Bramer, M., Wu, X., eds.: Proceedings of The Fourth IEEE International Conference on Data Mining (ICDM-04), Brighton, UK, IEEE Computer Society Press (November 1–4 2004) 170–177
8. Poole, D.: Probabilistic Horn abduction and Bayesian networks. Artificial Intelligence Journal **64** (1993) 81–129
9. Ngo, L., Haddawy, P.: Answering Queries from Context-Sensitive Probabilistic Knowledge Bases. Theoretical Computer Science **171** (1997) 147–177
10. Jäger, M.: Relational Bayesian Networks. In Laskey, K., Prade, H., eds.: Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI-97), Stockholm, Sweden, Morgan Kaufmann (July 30–August 1 1997) 266–273
11. Pfeffer, A.: Probabilistic Reasoning for Complex Systems. PhD thesis, Computer Science Department, Stanford University (December 2000)
12. Getoor, L.: Learning Statistical Models from Relational Data. PhD thesis, Stanford University, USA (June 2001)
13. Kersting, K., De Raedt, L.: Bayesian Logic Programming: Theory and Tool. In Getoor, L., Taskar, B., eds.: An Introduction to Statistical Relational Learning. MIT Press (2007) 291–321
14. Muggleton, S., De Raedt, L.: Inductive Logic Programming: Theory and Methods. Journal of Logic Programming **19**(20) (1994) 629–679
15. De Raedt, L.: Logical and Relational Learning. Springer (2008)
16. Džeroski, S., Lavrač, N., eds.: Relational data mining. Springer-Verlag, Berlin (2001)
17. Getoor, L., Taskar, B., eds.: Introduction to Statistical Relational Learning. The MIT Press (2007)
18. De Raedt, L., Frasconi, P., Kersting, K., Muggleton, S., eds.: Probabilistic Inductive Logic Programming. Volume 4911 of Lecture Notes in Computer Science. Springer (2008)
19. De Raedt, L., Kimmig, A., Toivonen, H.: Problog: A probabilistic Prolog and its application in link discovery. In Veloso, M., ed.: Proceedings of the 20th International Joint Conference on Artificial Intelligence. (2007) 2462–2467
20. Cussens, J.: Parameter estimation in stochastic logic programs. Machine Learning Journal **44**(3) (2001) 245–271
21. Poole, D.: The independent choice logic for modelling multiple agents under uncertainty. Artificial Intelligence **94**(1–2) (1997) 7–56
22. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society **B 39** (1977) 1–39
23. McLachlan, G., Krishnan, T.: The EM Algorithm and Extensions. Wiley, New York (1997)
24. Getoor, L., Taskar, B., Koller, D.: Using probabilistic models for selectivity estimation. In: Proceedings of ACM SIGMOD International Conference on Management of Data, ACM Press (2001) 461–472
25. Segal, E., Taskar, B., Gasch, A., Friedman, N., Koller, D.: Rich probabilistic models for gene expression. Bioinformatics **17**(Suppl 1) (2001) S243–52 Proc. ISMB 2001.
26. Segal, E., Battle, A., Koller, D.: Decomposing gene expression into cellular processes. In: Proc. Pacific Symposium on Biocomputing (PSB), World Scientific (January 2003) 89–100
27. Getoor, L., Rhee, J., Koller, D., Small, P.: Understanding tuberculosis epidemiology using probabilistic relational models. Journal of Artificial Intelligence in Medicine **30** (2004) 233–256

28. McGovern, A., Friedland, L., Hay, M., Gallagher, B., Fast, A., Neville, J., Jensen, D.: Exploiting relational structure to understand publication patterns in high-energy physics. SIGKDD Explorations **5**(2) (2003) 165–173
29. Neville, J., Simsek, Ö., Jensen, D., Komoroske, J., Palmer, K., Goldberg, H.: Using relational knowledge discovery to prevent securities fraud. In: Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press (2005)
30. Anguelov, D., Taskar, B., Chatalbashev, V., Koller, D., Gupta, D., Heitz, G., Ng, A.: Discriminative Learning of Markov Random Fields for Segmentation of 3D Scan Data. In Schmid, C., Soatto, S., Tomasi, C., eds.: IEEE Computer Society International Conference on Computer Vision and Pattern Recognition (CVPR-05). Volume 2., San Diego, CA, USA (June 20 – 26 2005) 169–176
31. Limketkai, B., Liao, L., Fox, D.: Relational Object Maps for Mobile Robots. In Giunchiglia, F., Kaelbling, L.P., eds.: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), Edinburgh, Scotland, AAAI Press (July 30,– August, 5 2005) 1471–1476
32. Kersting, K., De Raedt, L., Raiko, T.: Logial Hidden Markov Models. Journal of Artificial Intelligence Research (JAIR) **25** (2006) 425–456
33. Poon, H., Domingos, P.: Joint unsupervised coreference resolution with markov logic. In: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP), Honolulu, HI, USA (2008)
34. Xu, Z., Tresp, V., Rettinger, A., Kersting, K.: Social network mining with non-parametric relational models. In: Advances in Social Network Mining and Analysis. LNCS. Springer (2009)
35. Kersting, K., Xu, Z.: Learning preferences with hidden common cause relations. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 09). LNAI, Bled, Slovenia, Springer (Sept. 7-11 2009)
36. Poon, H., Domingos, P.: Unsupervised semantic parsing. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP), Singapore (2009)
37. Poole, D.: First-order probabilistic inference. In Gottlob, G., Walsh, T., eds.: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-03), Acapulco, Mexico, Morgan Kaufmann (August 2003) 985–991
38. de Salvo Braz, R., Amir, E., Roth, D.: Lifted First Order Probabilistic Inference. In: Proc. of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05). (2005) 1319–1325
39. Jaimovich, A., Meshi, O., Friedman, N.: Template-based inference in symmetric relational Markov random fields. In: Proc. of the Conf. on Uncertainty in Artificial Intelligence (UAI-07). (2007) 191–199
40. Milch, B., Zettlemoyer, L., Kersting, K., Haimes, M., Pack Kaelbling, L.: Lifted Probabilistic Inference with Counting Formulas. In: Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08). (July 13-17 2008)
41. Singla, P., Domingos, P.: Lifted First-Order Belief Propagation. In: Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08), Chicago, IL, USA (July 13-17 2008) 1094–1099
42. Sen, P., Deshpande, A., Getoor, L.: Exploiting Shared Correlations in Probabilistic Databases. In: Proc. of the Intern. Conf. on Very Large Data Bases (VLDB-08). (2008)

43. Kisynski, J., Poole, D.: Lifted aggregation in directed first-order probabilistic models. In Boutilier, C., ed.: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-09). (2009)
44. Kersting, K., Ahmadi, B., Natarajan, S.: Counting belief propagation. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI-09). (June 18–21 2009)
45. Kok, S., Domingos, P.: Statistical predicate invention. In: Proceedings of the Twenty-Fourth International Conference on Machine Learning (ICML-07), Corvallis, OR, USA, ACM Press (2007) 433–440
46. Gutmann, B., Kersting, K.: Tildecrf: Conditional random fields for logical sequences. In J. Fuernkranz, T. Scheffer, M.S., ed.: Proceedings of the 17th European Conference on Machine Learning (ECML-2006), Berlin, Germany (September 18-22 2006) 174–185
47. Chen, J., Muggleton, S., Santos, J.: Learning probabilistic logic models from probabilistic examples. Machine Learning **73**(1) (2008) 55–85
48. Passerini, A., Frasconi, P., De Raaedt, L.: Kernels on prolog proof trees: Statistical learning in the ilp setting. Journal of Machine Learning Research **7** (2006) 307–342
49. Milch, B., Marthi, B., Russell, S., Sontag, D., Ong, D., Kolobov, A.: BLOG: Probabilistic Models with Unknown Objects. In Giunchiglia, F., Kaelbling, L.P., eds.: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), Edinburgh, Scotland, AAAI Press (July 30,– August, 5 2005) 1352–1359
50. Chu, W., Sindhwani, V., Ghahramani, Z., Keerthi, S.: Relational learning with gaussian processes. In: Neural Information Processing Systems. (2006)
51. Silva, R., Chu, W., Ghahramani, Z.: Hidden common cause relations in relational learning. In: Neural Information Processing Systems. (2007)
52. Wang, J., Domingos, P.: Hybrid markov logic networks. In: Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI-08), Chicago, IL, USA (July 13-17 2008) 1106–1111
53. Xu, Z., Kersting, K., Tresp, V.: Multi-relational learning with gaussian processes. In Boutilier, C., ed.: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-09). (2009)
54. Xu, Z., Tresp, V., Yu, K., Kriegel, H.P.: Infinite hidden relational models. In: Proc. 22nd UAI. (2006)
55. Kemp, C., Tenenbaum, J., Griffiths, T., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: Proc. 21st AAAI. (2006)
56. Yu, K., Chu, W., Yu, S., Tresp, V., Xu, Z.: Stochastic relational models for discriminative link prediction. In: Neural Information Processing Systems. (2006)
57. Yu, K., Chu, W.: Gaussian process models for link analysis and transfer learning. In: Neural Information Processing Systems. (2007)
58. Singh, A., Gordon., G.: Relational learning via collective matrix factorization. In: Proc. 14th Intl. Conf. on Knowledge Discovery and Data Mining. (2008)