Data Mining – Statistical Relational Learning

Andreas Karwath   Jörg Wicker

Johannes Gutenberg University Mainz

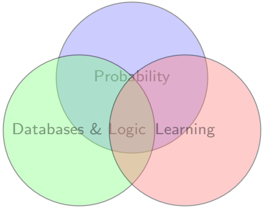
March 21, 2017

Overview

- Introduction
- Two types of semantics
- Problog

Objective

One of the key open questions in artificial intelligence concerns the integration of probabilistic reasoning, relational or logical representations and machine learning: statistical relational learning (SRL)



Why SRL?

- Structured domains
  - Not flat, but structured
  - Multi-relational, heterogeneous
  - semi-structured (?)
- Imperfect data
  - Noisy data
  - Missing data
  - Hidden variables
- Machine Learning
  - Knowledge acquisition bottleneck
  - cheap data (depending on domain)

Combining the (Dis-)Advantages

- Probabilistic logics
  - soft reasoning, expressiveness
  - no learning, but too expensive to handcraft models
- Statistical learning
  - soft reasoning, learning
  - propositional (attribute-value) representations: some learning problems cannot elegantly be described using attribute value representations
- Multi-relational data mining and inductive logic programming
  - sound logical reasoning
  - expressiveness and learning
  - some learning problems cannot elegantly be described without explicit handling of uncertainty

Why SRL?

- Rich probabilistic models
- Comprehensibility?
- Generalization to similar situations and similar individuals
- Parameter reduction (*parameter tying*)
- Learning: reuse of experience (training one random variable might improve the prediction for another random variable)

When to apply SRL?

- When it is not possible to represent a problem in propositional (attribute value) form easily
  - variable number of objects in examples
  - relations among objects are important
- Background knowledge can be defined intensionally (in the form of rules)

Two Types of Semantics[1]

The example refer to a domain with objects (birds,...) and the possibility of these objects to fly or not. In particular, there is one bird, tweety.

- Objective: sampling or frequentist view
  - "The probability that a randomly chosen bird will fly is greater than 0.9"
  - $w_x(\mathit{flies}(x)) > 0.9$
  - "One world"
- Subjective: degree-of-belief and based on possible worlds
  - "The probability that tweety flies is greater than 0.9"
  - $w(\mathit{flies}(\mathit{tweety})) > 0.9$

[1] Halpern, 1989

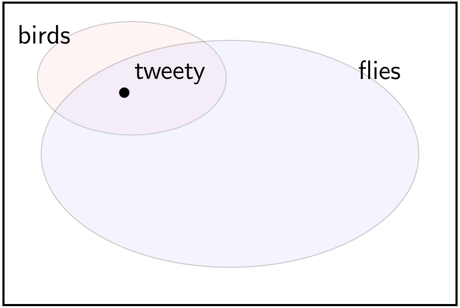
Sampling Probabilities

- A type-1 probability structure is a standard first-order structure with a probability distribution associated
- Randomness follows from a stochastic process such as randomly choosing a bird or throwing a dice
- E.g.,

```
0.10 : dice(1). 0.15 : dice(2).
0.10 : dice(3). 0.20 : dice(4).
0.25 : dice(5). 0.20 : dice(6).
```

Sampling Probabilities

As for tweety... (in our universe almost everything flies)

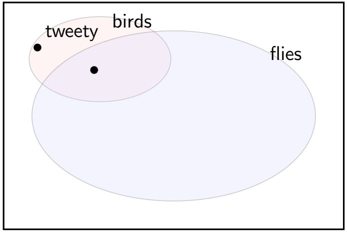


Degree of Belief Probabilities

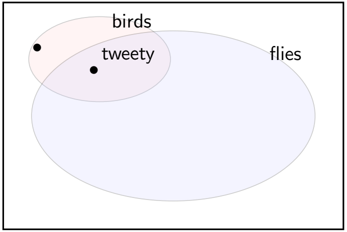
- A type-2 probability structure is a probability distribution *over possible worlds* (the worlds have probabilities!)
- Randomness follows from a lack of knowledge
- Suppose we have only two possible worlds, s_1 and s_2, with $\mu(s_1) = 0.2$ and $\mu(s_2) = 0.8$

Degree of Belief Probabilities

In s_1, tweety flies, in s_2 it does not:



$\mu(s_1) = 0.2$

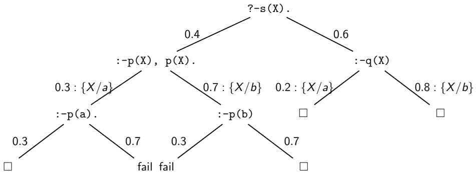


$\mu(s_2) = 0.8$

Stochastic Logic Programs in a Nutshell[2]

```
0.4:s(X) :- p(X), p(X)        0.3:p(a)    0.2:q(a)
0.6:s(X) :- q(X)              0.7:p(b)    0.8:q(b)
```

SLD-Tree:



$$
\begin{aligned}
p_\lambda(\texttt{s(a)}) &= (0.4 \times 0.3 \times 0.3 + 0.6 \times 0.2)/0.832 \\
&= 0.1875 \\
p_\lambda(\texttt{s(b)}) &= (0.4 \times 0.7 \times 0.7 + 0.6 \times 0.8)/0.832 \\
&= 0.8125
\end{aligned}
$$

[2] Muggleton, 1995

Problog - Example

```
0.8::edge(1,2).
0.7::edge(2,3).
0.6::edge(1,3).

path(X,Y) :- edge(X,Y).
path(X,Y) :- edge(X,Z),
             Y \== Z,
             path(Z,Y)).

query(path(1,3)).
```

Probability of Query Answer Given Program

A ProbLog program $T = \{p1 : c_1, \ldots, p_n : c_n\}$ defines a probability distribution over logic programs $L \subseteq L_T = \{c_1, \ldots, c_n\}$ in the following way:

$$
\begin{aligned}
P(L|T) &= \prod_{c_i \in L} p_i \prod_{c_i \in L_T \setminus L} (1 - p_i) \\
P(q|L) &= \begin{cases} 1 & \exists\theta : L \models q\theta \\ 0 & \text{otherwise} \end{cases} \\
P(q, L|T) &= P(q|L) \times P(L|T) \\
P(q|T) &= \sum_{M \subseteq L_T} P(q, M|T)
\end{aligned}
$$

Problog - Example

```
0.8::edge(1,2).
0.7::edge(2,3).
0.6::edge(1,3).

path(X,Y) :- edge(X,Y).
path(X,Y) :- edge(X,Z),
             Y \== Z,
             path(Z,Y)).

query(path(1,3)).
```

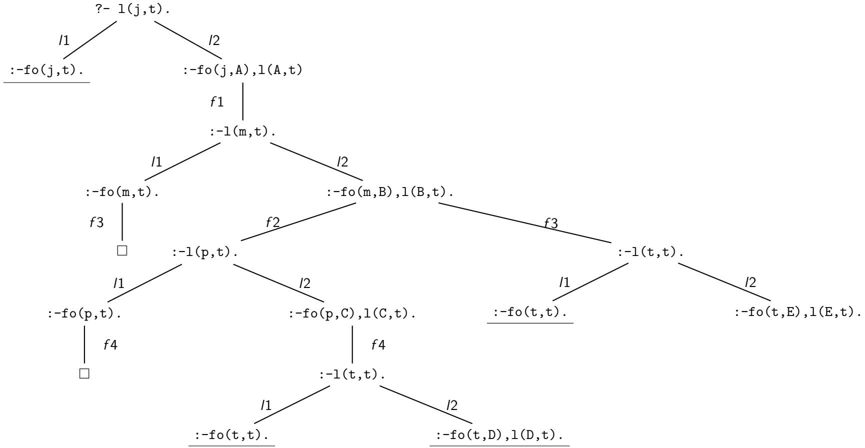
e(1,2)	e(2,3)	e(1,3)	
f	f	f	
f	f	t	$0.2 \times 0.3 \times 0.6$
f	t	f	
f	t	t	$0.2 \times 0.7 \times 0.6$
t	f	f	
t	f	t	$0.8 \times 0.3 \times 0.6$
t	t	f	$0.8 \times 0.7 \times 0.4$
t	t	t	$0.8 \times 0.7 \times 0.6$
			0.824

Problog - Example 2

Example knowledge base of probabilistic clauses and facts

```
1.0 :: likes(X, Y) :- friendof(X, Y).                %l1
0.8 :: likes(X, Y) :- friendof(X,Z), likes(Z, Y).    %l2
0.5 :: friendof(john,mary).                          %f1
0.5 :: friendof(mary,pedro).                         %f2
0.5 :: friendof(mary,tom).                           %f3
0.5 :: friendof(pedro,tom).                          %f4
```

Problog - Example 2
SLD-Tree of Query likes(john,tom)



Labels (propositional variable) indicates which clause is used in the proof (not which substitution).

Probability of Propositional DNF

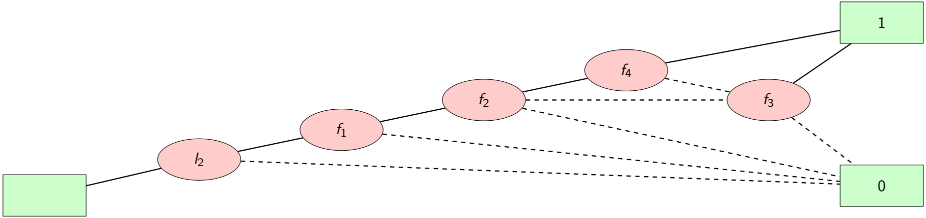
$$P(q|T) = P\left(\bigvee_{b \in pr(q)} \bigwedge_{b_i \in cl(b)}\right)$$

$$P(\texttt{likes(john,tom)}|T) \quad = \quad P((l_1 \wedge l_2 \wedge f_1 \wedge f_2 \wedge f_4) \vee (l_1 \wedge l_2 \wedge f_1 \wedge f_3))$$

Since $P(l_1) = 1$ this is equal to:

$$P(\texttt{likes(john,tom)}|T) \quad = \quad P((l_2 \wedge f_1 \wedge f_2 \wedge f_4) \vee (l_2 \wedge f_1 \wedge f_3))$$

Binary Decision Diagram (BDD) of the Example



Relationship between nodes from left to right
A child being above (high) from parent: *true*
A child being below (low) from parent: *false*

(Naive) Algorithm to Calculate Probabilities Based on BDD

Algorithm 1: $\textsc{Probability}$

Input: BDD node n

if *n is the 1-terminal node* **then**
    return *1*
end
if *n is the 0-terminal node* **then**
    return *0*
end
let h and l be the high and low children of n
return $p_n \times \textsc{Probability}(h) + (1 - p_n) \times \textsc{Probability}(l)$

Binary Decision Diagram (BDD) of the Example