Probabilistic Logic Programming: Semantics, Inference and Learning

Fabrizio Riguzzi

Department of Mathematics and Computer Science
University of Ferrara, Italy
fabrizio.riguzzi@unife.it

Joint work with Evelina Lamma, Theresa Swift, Elena Bellodi, Riccardo Zese, Arnaud Nguembang Fadja, Damiano Azzolini, Giuseppe Cota, Marco Alberti, Stefano Bragaglia

Dipartimento di Matematica e Informatica

Outline

Dipartimento di Matematica e Informatica

Outline

Dipartimento di Matematica e Informatica

Combining Logic and Probability

- Logic does not handle well uncertainty
- Graphical models do not handle well relationships among entities
- Solution: combine the two
- Many approaches proposed in the areas of Logic Programming, Uncertainty in AI, Machine Learning, Databases, Knowledge Representation

Probabilistic Logic Programming

- Distribution Semantics [Sato ICLP95]
- A probabilistic logic program defines a probability distribution over normal logic programs (called instances or possible worlds or simply worlds)
- The distribution is extended to a joint distribution over worlds and interpretations (or queries)
- The probability of a query is obtained from this distribution

Dipartimento di Matematica e Informatica

Probabilistic Logic Programming (PLP) Languages under the Distribution Semantics

- Probabilistic Logic Programs [Dantsin RCLP91]
- Probabilistic Horn Abduction [Poole NGC93], Independent Choice Logic (ICL) [Poole AI97]
- PRISM [Sato ICLP95]
- Logic Programs with Annotated Disjunctions (LPADs) [Vennekens et al. ICLP04]
- ProbLog [De Raedt et al. IJCAI07]
- They differ in the way they define the distribution over logic programs

Logic Programs with Annotated Disjunctions

$$
\begin{array}{l}
sneezing(X) : 0.7 \; ; \; null : 0.3 \leftarrow flu(X). \\
sneezing(X) : 0.8 \; ; \; null : 0.2 \leftarrow hay\_fever(X). \\
flu(bob). \\
hay\_fever(bob).
\end{array}
$$

- Distributions over the head of rules
- *null* does not appear in the body of any rule
- Worlds obtained by selecting one atom from the head of each grounding of each clause

ProbLog

$$
\begin{array}{l}
sneezing(X) \leftarrow flu(X), flu\_sneezing(X). \\
sneezing(X) \leftarrow hay\_fever(X), hay\_fever\_sneezing(X). \\
flu(bob). \\
hay\_fever(bob). \\
0.7 :: flu\_sneezing(X). \\
0.8 :: hay\_fever\_sneezing(X).
\end{array}
$$

- Distributions over facts
- Worlds obtained by selecting or not each grounding of each probabilistic fact

Distribution Semantics

- Case of no function symbols: finite Herbrand universe, finite set of groundings of each clause
- Atomic choice: selection of the i-th atom for grounding $C\theta$ of clause C
  - represented with the triple (C, θ, i)
- Example $C_1 = \mathit{sneezing}(X) : 0.7 \; ; \; \mathit{null} : 0.3 \leftarrow \mathit{flu}(X).$, $(C_1, \{X/\mathit{bob}\}, 1)$
- Composite choice κ: consistent set of atomic choices
- The probability of composite choice κ is

$$P(\kappa) = \prod_{(C,\theta,i)\in\kappa} P_0(C, i)$$

Distribution Semantics

- Selection σ: a total composite choice (one atomic choice for every grounding of each clause)
- A selection σ identifies a logic program w_σ called world
- The probability of w_σ is $P(w_\sigma) = P(\sigma) = \prod_{(C,\theta,i)\in\sigma} P_0(C,i)$
- Finite set of worlds: $W_\mathcal{P} = \{w_1, \ldots, w_m\}$
- $P(w)$ distribution over worlds: $\sum_{w\in W_\mathcal{P}} P(w) = 1$

Dipartimento di Matematica e Informatica

Distribution Semantics

- Ground query Q
- $P(Q|w) = 1$ if Q is true in w ($WFM(w) \models Q$) and 0 otherwise
- $P(Q) = \sum_w P(Q, w) = \sum_w P(Q|w)P(w) = \sum_{w \models Q} P(w)$

Dipartimento di Matematica e Informatica

Example Program (LPAD) Worlds

$$\begin{array}{ll}
sneezing(bob) \leftarrow flu(bob). & null \leftarrow flu(bob). \\
sneezing(bob) \leftarrow hay_fever(bob). & sneezing(bob) \leftarrow hay_fever(bob). \\
flu(bob). & flu(bob). \\
hay_fever(bob). & hay_fever(bob). \\
P(w_1) = 0.7 \times 0.8 & P(w_2) = 0.3 \times 0.8
\end{array}$$

$$\begin{array}{ll}
sneezing(bob) \leftarrow flu(bob). & null \leftarrow flu(bob). \\
null \leftarrow hay_fever(bob). & null \leftarrow hay_fever(bob). \\
flu(bob). & flu(bob). \\
hay_fever(bob). & hay_fever(bob). \\
P(w_3) = 0.7 \times 0.2 & P(w_4) = 0.3 \times 0.2
\end{array}$$

$$P(Q) = \sum_{w \in W_{\mathcal{P}}} P(Q, w) = \sum_{w \in W_{\mathcal{P}}} P(Q|w)P(w) = \sum_{w \in W_{\mathcal{P}} : w \models Q} P(w)$$

- $sneezing(bob)$ is true in 3 worlds
- $P(sneezing(bob)) = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94$

Dipartimento di Matematica e Informatica

Example Program (ProbLog) Worlds

- 4 worlds

$$
\begin{array}{l}
sneezing(X) \leftarrow flu(X), flu\_sneezing(X). \\
sneezing(X) \leftarrow hay\_fever(X), hay\_fever\_sneezing(X). \\
flu(bob). \\
hay\_fever(bob).
\end{array}
$$

$$
\begin{array}{ll}
flu\_sneezing(bob). & \\
hay\_fever\_sneezing(bob). & hay\_fever\_sneezing(bob). \\
P(w_1) = 0.7 \times 0.8 & P(w_2) = 0.3 \times 0.8 \\
flu\_sneezing(bob). & \\
P(w_3) = 0.7 \times 0.2 & P(w_4) = 0.3 \times 0.2
\end{array}
$$

$$P(Q) = \sum_{w \in W_{\mathcal{P}}} P(Q, w) = \sum_{w \in W_{\mathcal{P}}} P(Q|w)P(w) = \sum_{w \in W_{\mathcal{P}} : w \models Q} P(w)$$

- $sneezing(bob)$ is true in 3 worlds
- $P(sneezing(bob)) = 0.7 \times 0.8 + 0.3 \times 0.8 + 0.7 \times 0.2 = 0.94$

Logic Programs with Annotated Disjunctions

$$
\begin{array}{l}
strong\_sneezing(X) : 0.3 \; ; \; moderate\_sneezing(X) : 0.5 \leftarrow flu(X). \\
strong\_sneezing(X) : 0.2 \; ; \; moderate\_sneezing(X) : 0.6 \leftarrow hay\_fever(X). \\
flu(bob). \\
hay\_fever(bob).
\end{array}
$$

- 9 worlds
- $strong\_sneezing(bob)$ is true in 5
- $P(strong\_sneezing(bob)) = 0.3 \cdot 0.2 + 0.3 \cdot 0.6 + 0.3 \cdot 0.2 + 0.5 \cdot 0.2 + 0.2 \cdot 0.2 = 0.44$

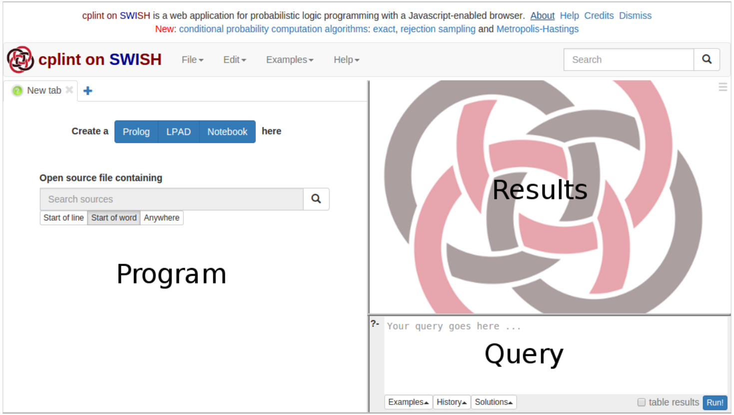
Dipartimento di Matematica e Informatica

Expressive Power

- All languages under the distribution semantics have the same expressive power
- LPADs have the most general syntax
- There are transformations that can convert each one into the others

cplint

- cplint system for inference and learning
- Web interface https://cplint.eu



Dipartimento di Matematica e Informatica

Outline

Dipartimento di Matematica e Informatica

Function Symbols

- What if function symbols are present?
- Infinite, denumerable Herbrand universe
- Infinite, denumerable Herbrand base
- Infinite, denumerable grounding of the program $\mathcal{P}$
- Each world infinite, denumerable
- $P(w) = 0$
- Uncountable $W_{\mathcal{P}}$
- Semantics not well-defined

Dipartimento di Matematica e Informatica

Game of Cards

$$
\begin{array}{l}
F_1 = 1/3 :: spades(X).\\
F_2 = 1/2 :: clubs(X).\\
pick(0, spades) \leftarrow spades(0).\\
pick(0, clubs) \leftarrow \sim spades(0), clubs(0).\\
pick(0, hearts) \leftarrow \sim spades(0), \sim clubs(0).\\
pick(s(X), spades) \leftarrow pick(X, \_), \sim pick(X, hearts), spades(s(X)).\\
pick(s(X), clubs) \leftarrow pick(X, \_), \sim pick(X, hearts), \sim spades(s(X)), clubs(s(X)).\\
pick(s(X), hearts) \leftarrow pick(X, \_), \sim pick(X, hearts), \sim spades(s(X)), \sim clubs(s(X)).\\
at\_least\_once\_spades \leftarrow pick(\_, spades).\\
never\_spades \leftarrow \sim at\_least\_once\_spades.
\end{array}
$$

Dipartimento di Matematica e Informatica

Function Symbols

- The set of worlds ω_κ compatible with a composite choice κ is $\omega_\kappa = \{w_\sigma \in W_\mathcal{P} | \kappa \subseteq \sigma\}$.
- For programs without function symbols, $P(\kappa) = \sum_{w \in \omega_\kappa} P(w)$
- For program with function symbols $\sum_{w \in \omega_\kappa} P(w)$ may not be defined as ω_κ may be uncountable and $P(w) = 0$.
- $P(\kappa)$ is still well defined. Let us call it μ so $\mu(\kappa) = P(\kappa)$.

Dipartimento di Matematica e Informatica

Function Symbols

- Given a *set* of composite choices K, the set of worlds ω_K compatible with K is $\omega_K = \bigcup_{\kappa \in K} \omega_\kappa$.
- Two composite choices κ_1 and κ_2 are incompatible if their union is not consistent.
- A set K of composite choices is pairwise incompatible if for all $\kappa_1 \in K, \kappa_2 \in K$, $\kappa_1 \neq \kappa_2$ implies that κ_1 and κ_2 are incompatible.

Function Symbols

- The probability of a pairwise incompatible set K of composite choices can be defined as $P(K) = \sum_{\kappa \in K} P(\kappa)$
- $\mu(K) = P(K)$
- Two sets K_1 and K_2 of composite choices are equivalent if they correspond to the same set of worlds: $\omega_{K_1} = \omega_{K_2}$.
- Given a query q, a composite choice κ is an explanation for q if $\forall w \in \omega_\kappa : w \models q$.
- A set K of composite choices is covering with respect to q if every world in which q is true belongs to ω_K.

Dipartimento di Matematica e Informatica

Game of Cards

- The set $K = \{\kappa_1, \kappa_2\}$ with

$$\kappa_1 = \{(f_1, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1)\}$$
$$\kappa_2 = \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1)\}$$

  is a pairwise incompatible finite set of finite explanations that are covering for the query $pick(s(0), spades)$

- $P(on(s(0), 1)) = P(K) = 1/3 \cdot 1/3 + 2/3 \cdot 1/2 \cdot 1/3 = 2/9$

Dipartimento di Matematica e Informatica

Function Symbols

Theorem (Existence of a pairwise incompatible set of composite choices (Poole JLP00))

Given a finite set K of composite choices, there exists a finite set K' of pairwise incompatible composite choices such that K and K' are equivalent.

Theorem (Equivalence of the probability of two equivalent pairwise incompatible finite sets of finite composite choices (Poole AI03))

If K_1 and K_2 are both pairwise incompatible finite sets of finite composite choices such that they are equivalent, then $P(K_1) = P(K_2)$.

Probability Measure

For a probabilistic logic program $\mathcal{P}$, we can define the probability measure $\mu_{\mathcal{P}} : \Omega_{\mathcal{P}} \to [0,1]$ where $\Omega_{\mathcal{P}}$ is defined as the set of sets of worlds identified by countable sets of countable composite choices: $\Omega_{\mathcal{P}} = \{\omega_K | K$ is a countable set of countable composite choices $\}$.

Theorem (σ-algebra of a program)

$\Omega_{\mathcal{P}}$ is an σ-algebra over $W_{\mathcal{P}}$.

Function Symbols

Theorem (Probability space of a program)

The triple $\langle W_{\mathcal{P}}, \Omega_{\mathcal{P}}, \mu_{\mathcal{P}} \rangle$ *with*

$$\mu_{\mathcal{P}}(\omega_K) = \lim_{n \to \infty} \mu(K'_n)$$

where $K = \{\kappa_1, \kappa_2, \ldots\}$ *and* K'_n *is a pairwise incompatible set of composite choices equivalent to* $\{\kappa_1, \ldots, \kappa_n\}$, *is a probability space*

Example

The query *at_least_once_spades* has the pairwise incompatible covering set of explanations $K^+ = \{\kappa_0^+, \kappa_1^+, \dots\}$ with

$$\begin{array}{rcl}
\kappa_0^+ & = & \{(f_1, \{X/0\}, 1)\} \\
\kappa_1^+ & = & \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), (f_1, \{X/s(0)\}, 1)\} \\
\dots & & \\
\kappa_i^+ & = & \{(f_1, \{X/0\}, 0), (f_2, \{X/0\}, 1), \dots, (f_1, \{X/s^{i-1}(0)\}, 0), \\
 & & (f_2, \{X/s^{i-1}(0)\}, 1), (f_1, \{X/s^i(0)\}, 1)\} \\
\dots & &
\end{array}$$

$$\begin{array}{rcl}
P(at_least_once_spades) & = & \frac{1}{3} + \frac{2}{3} \cdot \frac{1}{2} \cdot \frac{1}{3} + \left(\frac{2}{3} \cdot \frac{1}{2}\right)^2 \cdot \frac{1}{3} + \dots \\
 & = & \frac{1}{3} + \frac{1}{9} + \frac{1}{27} \dots = \frac{\frac{1}{3}}{1 - \frac{1}{3}} = \frac{\frac{1}{3}}{\frac{2}{3}} = \frac{1}{2}
\end{array}$$

Dipartimento di Matematica e Informatica

Function Symbols

Theorem (Well-definedness of the distribution semantics (Riguzzi IJAR16))

For a sound ground probabilistic logic program $\mathcal{P}$, $\mu_{\mathcal{P}}(\{w|w \in W_{\mathcal{P}}, w \vDash a\})$ *for all* $a \in \mathcal{B}_{\mathcal{P}}$ *is well defined.*

Dipartimento di Matematica e Informatica

Continuous Random Variables

$p(X) : gaussian(X, 0, 1).$
$a \leftarrow p(X), X > 3$

- X follows a Gaussian distribution with mean 0 and variance 1
- a is true if X is greater than 3
- Constraints on random variables' range
- Probabilistic Constraint Logic Programs (Michels et al AI15)
- How about continuous random variables and function symbols?

Dipartimento di Matematica e Informatica

Continuous Random Variables and Function Symbols

- Variation of the previous program, with another requirement: the player spins a wheel, and the game continues only if the axis is in the range $]\pi, 2\pi]$

$\ldots$

$angle(\_, X) : uniform\_dens(X, 0, 6.28)$

$pick(0, spades) \leftarrow spades(0), angle(0, V), V > 3.14.$

$\ldots$

$pick(s(X), spades) \leftarrow pick(X, \_), \sim pick(X, hearts), spades(s(X)), angle(s(X), V), V > 3.14.$

$\ldots$

$at\_least\_once\_spades \leftarrow pick(\_, spades).$

$never\_spades \leftarrow \sim at\_least\_once\_spades.$

Function Symbols

Theorem (Well-definedness of the distribution semantics - PCLP (Azzolini, Riguzzi, Lamma AI21))

For a sound ground probabilistic constraint logic program $\mathcal{P}$, for all ground atoms a, $\mu_{\mathcal{P}}(\{w \mid w \in W_{\mathcal{P}}, WFM(w) \models a\})$ is well-defined.

Dipartimento di Matematica e Informatica

Outline

Dipartimento
di Matematica
e Informatica

Inference for PLP under DS

- Computing the probability of a query (no evidence)
- Knowledge compilation:
  - compile the program to an intermediate representation
    - Binary Decision Diagrams (ProbLog [De Raedt et al. IJCAI07], `cplint` [Riguzzi AIIA07,Riguzzi LJIGPL09], PITA [Riguzzi & Swift ICLP10, ICLP11])
    - deterministic Decomposable Negation Normal Form circuits (d-DNNF) (ProbLog2 [Fierens et al. TPLP15])
    - Sentential Decision Diagrams (ProbLog2 [Fierens et al. TPLP15])
  - compute the probability by weighted model counting

Dipartimento di Matematica e Informatica

Knowledge Compilation

- Assign Boolean random variables to the probabilistic rules
- Given a query Q, compute a covering set of explanation K
- Build the formula

$$F(Q) = \bigvee_{\kappa \in K} \bigwedge_{X \in \kappa} X \bigwedge_{\overline{X} \in \kappa} \overline{X}$$

- Build a BDD representing $F(Q)$

Dipartimento di Matematica e Informatica

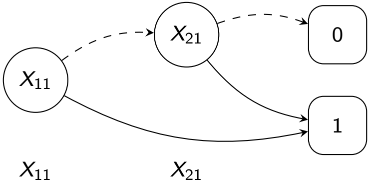
Example

A covering set of explanations for *sneezing*(*bob*) is $K = \{\kappa_1, \kappa_2\}$
$\kappa_1 = \{X_{11}\} \quad \kappa_2 = \{X_{21}\}$
$X_{11} \leftarrow (C_1, \theta_1 = \{X/bob\}, 1) \quad X_{21} \leftarrow (C_2, \theta_1 = \{X/bob\}, 1)$
$f_K(\mathbf{X}) = X_{11} \vee X_{21}$.

- In order to compute the probability, we must make the explanations mutually exclusive
- [De Raedt at. IJCAI07]: Binary Decision Diagram (BDD)

Binary Decision Diagrams

- A BDD for a function of Boolean variables is a rooted graph that has one level for each Boolean variable
- A node n in a BDD has two children: one corresponding to the 1 value of the variable associated with n and one corresponding the 0 value of the variable
- The leaves store either 0 or 1.



Tabling

- PITA (Probabilistic Inference with Tabling and Answer subsumption) (Riguzzi Swift ICLP 2010 ICLP11)
- All the explanations for a goal have to be found
- It makes sense to store the explanations for subgoals with tabling
- Associate to each answer (ground atom) a BDD representing its explanations
- Combine BDDs by using the Boolean operators offered by BDD manipulating packages
- Library for manipulating BDD directly in Prolog (interface to CUDD)
- A BDD is represented in Prolog by an integer indicating the address of its root node
- Casting for integer-pointer conversion

Dipartimento di Matematica e Informatica

Tabling

- Add an extra argument to each atom for storing a BDD
- When an answer $p(\mathbf{x}, bdd)$ is found, bdd represents the explanations for $p(\mathbf{x})$
- If the program is range restricted, $p(\mathbf{x})$ is ground
- Use program transformation to obtain a Prolog program from an LPAD

Dipartimento di Matematica e Informatica

Outline

Dipartimento
di Matematica
e Informatica

Approximate Inference

- Inference problem is #P hard
- For large models inference is intractable
- Approximate inference
  - Monte Carlo: draw samples of the truth value of the query
  - Iterative deepening: gives a lower and an upper bound
  - Compute only the best k explanations: branch and bound, gives a lower bound

Dipartimento di Matematica e Informatica

Monte Carlo - MCINTYRE (Riguzzi FI13)

- The disjunctive clause
  $C_r = H_1 : \alpha_1 \vee \ldots \vee H_n : \alpha_n \leftarrow L_1, \ldots, L_m.$
  is transformed into the set of clauses $MC(C_r)$
  $MC(C_r, 1) = H_1 \leftarrow L_1, \ldots, L_m, sample\_head(r, Vars, [\alpha_1, \ldots, \alpha_n], NH), NH = 1.$
  $\ldots$
  $MC(C_r, n) = H_n \leftarrow L_1, \ldots, L_m, sample\_head(r, Vars, [\alpha_1, \ldots, \alpha_n], NH), NH = n.$
- Sample truth value of query `Q`:

```
...
  (call(Q)-> NT1 is NT+1; NT1 =NT),
...
```

Monte Carlo - MCINTYRE

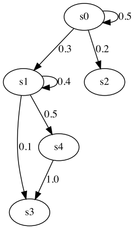
```
sample_head(R,Vars,_HeadList,N):-
  sampled(R,Vars,N),!.

sample_head(R,Vars,HeadList,N):-
  sample(HeadList,N),
  assertz(sampled(R,Vars,N)).
```

- Simplicity of implementation
- The estimate can be improved as more time is available, making it an anytime algorithm.

Dipartimento di Matematica e Informatica

Markov Chain Example:

Model checking of a Markov chain: we want to know what is the likelihood that on an execution of the chain from a start state s, a final state t will be reached?



- The chains may be infinite so the query may have an infinite number of explanations
- PITA may not terminate.
- Two solutions. We may either fix a bound on the depth of the derivations of PITA by setting the parameters

```
:- set_pita(depth_bound,true).
:- set_pita(depth,<level of depth (integer)>).
```

Alternatively, MCINTYRE can be used.

Dipartimento di Matematica e Informatica

Monte Carlo for Hybrid Programs

- Monte Carlo inference can be used almost directly for approximate inference for hybrid programs.

$$C_i = g(X, Y) : gaussian(Y, 0, 1) \leftarrow object(X).$$

- MCINTYRE transforms it into (Riguzzi Bellodi Lamma Zese Cota SPE16, Alberti, Bellodi, Cota, Riguzzi, Zese IA17)

$$g(X, Y) \leftarrow object(X), sample\_gauss(i, [X], 0, 1, Y).$$

```
sample_gauss(R,Vars,_Mean,_Variance,S):-
  sampled(R,Vars,S),!.

sample_gauss(R,Vars,Mean,Variance,S):-
  gauss(Mean,Variance,S),
  assertz(sampled(R,Vars,S)).
```

Conditional Inference

- Computing the probability of a query given evidence: rejection sampling, Metropolis-Hastings or Gibbs Markov chain Monte Carlo.
- Rejection sampling: the evidence is first queried and, if it is successful, the query is asked in the same sample; otherwise, the sample is discarded.
- In Metropolis-Hastings and Gibbs MCMC, a Markov chain is built by taking an initial sample and by generating successor samples
- `cplint` implements both Metropolis-Hastings (Alberti, Bellodi, Cota, Riguzzi, Zese IA17) and Gibbs (Azzolini, Riguzzi, Lamma PLP20)

Dipartimento di Matematica e Informatica

Outline

Dipartimento di Matematica e Informatica

Parameter Learning

- Problem: given a set of interpretations, a program, find the parameters maximizing the likelihood of the interpretations (or of instances of a target predicate)
- The interpretations record the truth value of ground atoms, not of the choice variables
- Unseen data: relative frequency can't be used

Dipartimento
di Matematica
e Informatica

Parameter Learning

- UW-CSE domain (Kok, Domingos ICML05)
- The objective is to predict the “advised by” relation between students and professors.

$taughtby(ai, course44, person171).$
$taughtby(ai, course24, person240).$
$\ldots$
$courselevel(ai, course52, level\_400).$
$courselevel(ai, course44, level\_400).$
$\ldots$
$hasposition(ai, person292, faculty\_affiliate).$
$hasposition(ai, person293, faculty\_affiliate).$
$\ldots$
$advisedby(ai, person265, person168).$
$advisedby(ai, person381, person168).$
$\ldots$

Dipartimento di Matematica e Informatica

Parameter Learning

$$advisedby(A, B) : 0.53 \; ; \; tempadvisedby(A, B) : 0.26 \leftarrow ta(C, A), taughtby(C, B).$$
$$advisedby(A, B) : 0.03 \leftarrow publication(C, B), publication(C, A), professor(B), student(A).$$
$$advisedby(A, B) : 0.05 \leftarrow professor(B).$$
$$hasposition(A, faculty) : 0.34 \; ; \; hasposition(A, faculty\_adjunct) : 0.22 \; ;$$
$$hasposition(A, faculty\_emeritus) : 0.14 \; ;$$
$$hasposition(A, faculty\_visiting) : 0.09 \leftarrow professor(A).$$
$$professor(A) : 0.56 \leftarrow hasposition(A, B).$$
$$\ldots$$

Dipartimento di Matematica e Informatica

Parameter Learning

- An Expectation-Maximization algorithm must be used:
  - Expectation step: the distribution of the unseen variables in each instance is computed given the observed data
  - Maximization step: new parameters are computed from the distributions using relative frequency
  - End when likelihood does not improve anymore

Dipartimento di Matematica e Informatica

EMBLEM (Bellodi Riguzzi IDA13)

- EM over Bdds for probabilistic Logic programs Efficient Mining
- Input: an LPAD; logical interpretations (data); *target* predicate(s)
- All ground atoms in the interpretations for the target predicate(s) correspond to as many queries
- BDDs encode the explanations for each query Q
- Expectations computed with two passes over the BDDs

Dipartimento di Matematica e Informatica

Outline

Dipartimento di Matematica e Informatica

Structure Learning for LPADs

- Given a trivial LPAD or an empty one, a set of interpretations (data)
- *Find the model and the parameters* that maximize the probability of the data (log-likelihood)
- SLIPCOVER: Structure LearnIng of Probabilistic logic program by searching OVER the clause space (Riguzzi Bellodi TPLP15)
  1. Beam search in the space of clauses to find the promising ones
  2. Greedy search in the space of probabilistic programs guided by the LL of the data.
- *Parameter learning* by means of EMBLEM

SLIPCOVER

- Cycle on the set of predicates that can appear in the head of clauses, either target or background
- For each predicate, beam search in the space of clauses
- The initial set of beams is generated by building a set of *bottom clauses* as in Progol (Muggleton NGC95)

Dipartimento di Matematica e Informatica

SLIPCOVER

- After the clause search phase, SLIPCOVER performs a greedy search in the space of theories:
  - it starts with an empty theory and adds a target clause at a time from the list TC.
  - After each addition, it runs EMBLEM and computes the LL of the data as the score of the resulting theory.
  - If the score is better than the current best, the clause is kept in the theory, otherwise it is discarded.
- Finally, SLIPCOVER adds all the background clauses to the theory and performs parameter learning on the resulting theory.

Dipartimento di Matematica e Informatica

Outline

Dipartimento di Matematica e Informatica

Scaling PILP

- PILP requires expensive learning procedures due to the high cost of inference.
- Two systems that try to remedy this are LIFTCOVER (Nguembang Fadja, Riguzzi ML19) and SLEAHP (Nguembang Fadja, Riguzzi, Lamma ML21)

Dipartimento di Matematica e Informatica

LIFTCOVER

- Lifted inference for reasoning on whole populations of individuals instead of considering each individual separately.
- Simple PLP language (liftable PLP) where programs contain clauses with a single annotated atom in the head and the predicate of this atom is the same for all clauses.
- In this case, all the approaches for lifted inference coincide and reduce to a simple computation.

UW-CSE

$advisedby(A, B) : 0.4 \leftarrow$
$\quad student(A), professor(B), publication(C, A), publication(C, B).$
$advisedby(A, B) : 0.5 \leftarrow$
$\quad student(A), professor(B), ta(C, A), taughtby(C, B).$

- The probability that a student is advised by a professor depends on the number of joint publications and the number of courses the professor teaches where the student is a TA, the higher these numbers the higher the probability.
- $q = advisedby(harry, ben)$ where *harry* is a student, *ben* is a professor, they have 4 joint publications and *ben* teaches 2 courses where *harry* is a TA.
  $P(advisedby(harry, ben)) = 1 - (1 - 0.4)^4(1 - 0.5)^2 = 0.9676.$

SLEAHP

- SLEAHP (Nguembang Fadja, Riguzzi, Lamma ML21) extends liftable PLP to Hierarchical Probabilistic Logic Programs (HPLPs)
- The computation of probabilities in such programs is truth-functional
- Independent-or assumption
- Suitable for domains where entities may be related to a varying number of other entities.

Dipartimento di Matematica e Informatica

UW-CSE

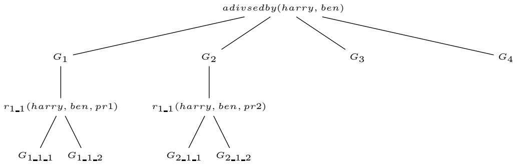
$$C_1 = advised\_by(A, B) : 0.3 \leftarrow$$
$$student(A), professor(B), project(C, A), project(C, B),$$
$$r_{1\_1}(A, B, C).$$
$$C_2 = advised\_by(A, B) : 0.6 \leftarrow$$
$$student(A), professor(B), ta(C, A), taughtby(C, B).$$
$$C_{1\_1\_1} = r_{1\_1}(A, B, C) : 0.2 \leftarrow$$
$$publication(P, A, C), publication(P, B, C).$$

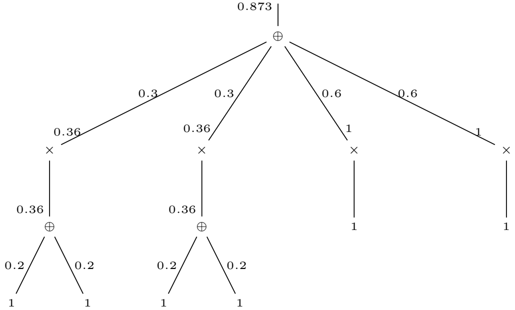
- The probability of $q = advised\_by(harry, ben)$ depends not only on the number of joint courses and projects but also on the number of joint publications from projects.

Dipartimento di Matematica e Informatica

UW-CSE

- *harry* and *ben* have two joint courses c_1 and c_2, two joint projects pr_1 and pr_2, two joint publications p_1 and p_2 from project pr_1 and two joint publications p_3 and p_4 from project pr_2.





- $p \oplus q = 1 - (1 - p) \cdot (1 - q)$

Outline

Dipartimento
di Matematica
e Informatica

Applications

- UW-CSE
- Mutagenesis (Srinivasan et al. AI96): quantitative structure-activity relationship (QSAR): predicting the biological activity of chemicals from their physicochemical properties or molecular structure.
- Carcinogenesis (Srinivasan et al. ILP97): QSAR, predict the cancerogenicity of compounds from their chemical structure.
- Mondial (Schulte, Khosravi ML12): information regarding geographical regions of the world
- Hepatitis (Khosravi et al. ML12): Discovery Challenge of ECML/PKDD 2002, information on laboratory examinations of hepatitis B and C infected patients
- Bupa (McDermot, Forsyth PRL16): diagnosing patients with liver disorders.

Dipartimento di Matematica e Informatica

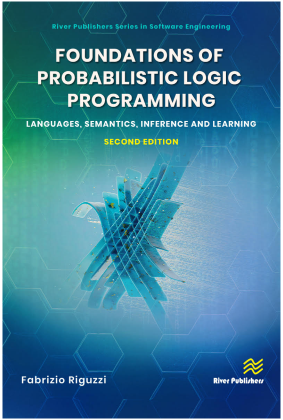
Applications

- NBA (Schulte, Routley, CIDM14): predicting the results of basketball matches from NBA.
- Pyrimidine, Triazine (Layne, Qiu, 2005): predicting the inhibition of dihydrofolate reductase by pyrimidines and triazines
- Financial (Berka ECMLDC00): predicting the success of loan applications by clients of a bank.
- Sisyphus (Blockeel, Struyf IDDM01): classifying households and persons in relation to private life insurance.
- Yeast (Davis et al. ECML05): predicting whether a yeast gene codes for a protein involved in metabolism.
- Event Calculus (Schwitter ICLP17): learning effect axioms for the Event Calculus

Dipartimento di Matematica e Informatica

Conclusions

- Probabilistic Logic Progrramming
- Semantics for programs with function symbols and continuous random variables
- Inference
- Learning
- Open problems
  - Exact inference with continuous variables
  - Learning for hybrid programs
  - Combining Deep Learning with PLP



out in October 2022

Dipartimento di Matematica e Informatica