Reasoning with Probabilistic Logic Programming Languages

Fabrizio Riguzzi

Department of Mathematica and Computer Science – University of Ferrara
fabrizio.riguzzi@unife.it

Dipartimento
di Matematica
e Informatica

Outline

- Exact inference
- Approximate inference
- Parameter learning
- Structure learning

Inference for PLP under DS

- EVID: compute an unconditional probability $P(e)$, the probability of evidence (also query in this case).
- COND: compute the conditional probability distribution of the query given the evidence, i.e. compute $P(q|e)$
- MPE or *most probable explanation*: find the most likely value of all non-evidence atoms given the evidence, i.e. solving the optimization problem $\arg\max_q P(q|e)$
- MAP or *maximum a posteriori*: find the most likely value of a set of non-evidence atoms given the evidence, i.e. finding $\arg\max_q P(q|e)$. MPE is a special case of MAP where $Q \cup E = H_T$.
- DISTR: compute the probability distribution or density of the non-ground arguments of a conjunction of literals q, e.g., computing the probability density of X in goal $mix(X)$ of the Gaussian mixture

Weight Learning

- Given
  - model: a probabilistic logic model with unknown parameters
  - data: a set of interpretations
- Find the values of the parameters that maximize the probability of the data given the model
- Discriminative learning: maximize the conditional probability of a set of outputs (e.g. ground instances for a predicate) given a set of inputs
- Alternatively, the data are queries for which we know the probability: minimize the error in the probability of the queries that is returned by the model

Structure Learning

- Given
  - language bias: a specification of the search space
  - data: a set of interpretations
- Find the formulas and the parameters that maximize the likelihood of the data given the model
- Discriminative learning: again maximize the conditional likelihood of a set of outputs given a set of inputs

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 (BDD) (ProbLog [De Raedt et al. IJCAI07], `cplint` [Riguzzi AIIA07,Riguzzi LJIGPL09], PITA [Riguzzi & Swift ICLP10])
    - deterministic, Decomposable Negation Normal Form circuit (d-DNNF) (ProbLog2 [Fierens et al. TPLP15])
    - Sentential Decision Diagrams
  - compute the probability by weighted model counting

Dipartimento
di Matematica
e Informatica

Inference for PLP under DS

- Bayesian Network based:
  - Convert to BN
  - Use BN inference algorithms (CVE [Meert et al. ILP09])
- Lifted inference

Knowledge Compilation

- Assign Boolean random variables to the probabilistic rules
- Given a query Q, compute its explanations, assignments to the random variables that are sufficient for entailing the query
- Let K be the set of all possible explanations
- Build a Boolean formula $F(Q)$
- Build a BDD representing $F(Q)$

Dipartimento di Matematica e Informatica

ProbLog

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

Definitions

- Composite choice κ: consistent set of atomic choices (C_i, θ_j, l) with $l \in \{1, 2\}$
- Set of worlds compatible with κ: $\omega_\kappa = \{w_\sigma | \kappa \subseteq \sigma\}$
- Explanation κ for a query Q: Q is true in every world of ω_κ
- A set of composite choices K is covering with respect to Q: every world w in which Q is true is such that $w \in \omega_K$ where $\omega_K = \bigcup_{\kappa \in K} \omega_\kappa$
- Example:

$$K_1 = \{\{(C_1, \{X/bob\}, 1)\}, \{(C_2, \{X/bob\}, 1)\}\} \quad (1)$$

is covering for *sneezing*(*bob*).

Dipartimento di Matematica e Informatica

Finding Explanations

- All explanations for the query are collected
- ProbLog: source to source transformation for facts, use of dynamic database
- `cplint` (PITA): source to source transformation, addition of an argument to predicates

Dipartimento di Matematica e Informatica

Explanation Based Inference Algorithm

- $K =$ set of explanations found for Q, the probability of Q is given by the probability of the formula

$$f_K(\mathbf{X}) = \bigvee_{\kappa \in K} \bigwedge_{(C_i,\theta_j,l) \in \kappa} (X_{C_i\theta_j} = l)$$

  where $X_{C_i\theta_j}$ is a random variable whose domain is $1, 2$ and $P(X_{C_i\theta_j} = l) = P_0(C_i, l)$
- Binary domain: we use a Boolean variable X_{ij} to represent $(X_{C_i\theta_j} = 1)$
- $\overline{X_{ij}}$ represents $(X_{C_i\theta_j} = 2)$

Dipartimento di Matematica e Informatica

Example

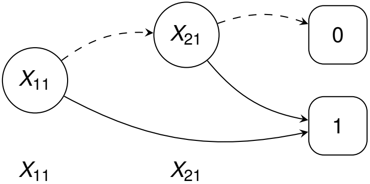
A set of covering explanations for $sneezing(bob)$ is $K = \{\kappa_1, \kappa_2\}$
$\kappa_1 = \{(C_1, \{X/bob\}, 1)\} \quad \kappa_2 = \{(C_2, \{X/bob\}, 1)\}$
$K = \{\kappa_1, \kappa_2\}$
$f_K(\mathbf{X}) = (X_{C_1\{X/bob\}} = 1) \vee (X_{C_2\{X/bob\}} = 1)$.
$X_{11} = (X_{C_1\{X/bob\}} = 1) \quad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

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

Dipartimento di Matematica e Informatica

Binary Decision Diagrams

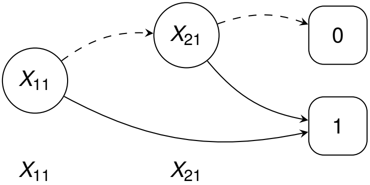
- A BDD for a function of Boolean variables is a rooted graph that has one level for each Boolean variable
- A node *n* in a BDD has two children: one corresponding to the 1 value of the variable associated with *n* and one corresponding the 0 value of the variable
- The leaves store either 0 or 1.



Binary Decision Diagrams

- BDDs can be built by combining simpler BDDs using Boolean operators
- While building BDDs, simplification operations can be applied that delete or merge nodes
- Merging is performed when the diagram contains two identical sub-diagrams
- Deletion is performed when both arcs from a node point to the same node
- A reduced BDD often has a much smaller number of nodes with respect to the original BDD

Binary Decision Diagrams



$$f_K(\mathbf{X}) = X_{11} \times f_K^{X_{11}}(\mathbf{X}) + \overline{X_{11}} \times f_K^{\overline{X_{11}}}(\mathbf{X})$$

$$P(f_K(\mathbf{X})) = P(X_{11})P(f_K^{X_{11}}(\mathbf{X})) + (1 - P(X_{11}))P(f_K^{\overline{X_{11}}}(\mathbf{X}))$$

$$P(f_K(\mathbf{X})) = 0.7 \cdot P(f_K^{X_{11}}(\mathbf{X})) + 0.3 \cdot P(f_K^{\overline{X_{11}}}(\mathbf{X}))$$

Dipartimento di Matematica e Informatica

Probability from a BDD

- Dynamic programming algorithm [De Raedt et al IJCAI07]
- Initialize map p; Call $\mathrm{Prob}(root)$
- Function $\mathrm{Prob}(n)$
- if $p(n)$ exists, return $p(n)$
- if n is a terminal note
  - return $value(n)$
- else
  - $prob := \mathrm{Prob}(child_1(n)) \times p(v(n)) + \mathrm{Prob}(child_0(n)) \times (1 - p(v(n)))$
  - Add $(n, prob)$ to p; return $prob$

Dipartimento di Matematica e Informatica

Logic Programs with Annotated Disjunctions

$$
\begin{array}{rlcl}
C_1 = & strong\_sneezing(X) : 0.3 \vee moderate\_sneezing(X) : 0.5 & \leftarrow & flu(X). \\
C_2 = & strong\_sneezing(X) : 0.2 \vee moderate\_sneezing(X) : 0.6 & \leftarrow & hay\_fever(X). \\
C_3 = & flu(bob). & & \\
C_4 = & hay\_fever(bob). & &
\end{array}
$$

- Distributions over the head of rules
- More than two head atoms

Dipartimento di Matematica e Informatica

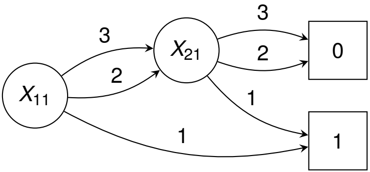
Example

A set of covering explanations for *strong_sneezing*(*bob*) is
$K = \{\kappa_1, \kappa_2\}$
$\kappa_1 = \{(C_1, \{X/bob\}, 1)\}$
$\kappa_2 = \{(C_2, \{X/bob\}, 1)\}$
$X_{11} = X_{C_1\{X/bob\}}$
$X_{21} = X_{C_2\{X/bob\}}$
$f_K(\mathbf{X}) = (X_{11} = 1) \vee (X_{21} = 1)$.
$P(f_X) = P(X_{11} = 1) + P(X_{21} = 1) - P(X_{11} = 1)P(X_{21} = 1)$

- To make the explanations mutually exclusive: Multivalued Decision Diagram (MDD)

Dipartimento di Matematica e Informatica

Multivalued Decision Diagrams



$$f_K(\mathbf{X}) = \bigvee_{l \in |X_{11}|} (X_{11} = l) \wedge f_K^{X_{11}=l}(\mathbf{X})$$

$$P(f_K(\mathbf{X})) = \sum_{l \in |X_{11}|} P(X_{11} = l) P(f_K^{X_{11}=l}(\mathbf{X}))$$

$$f_K(\mathbf{X}) = (X_{11} = 1) \wedge f_K^{X_{11}=1}(\mathbf{X}) + (X_{11} = 2) \wedge f_K^{X_{11}=2}(\mathbf{X}) + (X_{11} = 3) \wedge f_K^{X_{11}=3}(\mathbf{X})$$

$$f_K(\mathbf{X}) = 0.3 \cdot P(f_K^{X_{11}=1}(\mathbf{X})) + 0.5 \cdot P(f_K^{X_{11}=2}(\mathbf{X})) + 0.2 \cdot P(f_K^{X_{11}=3}(\mathbf{X}))$$

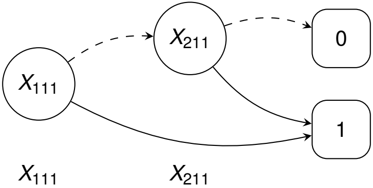
Manipulating Multivalued Decision Diagrams

- Use an MDD package
- Convert to BDD, use a BDD package: BDD packages more developed, more efficient
- Conversion to BDD
  - Log encoding
  - Binary splits: more efficient

Dipartimento di Matematica e Informatica

Transformation to a Binary Decision Diagram

- For a variable X_{ij} having n values, we use $n-1$ Boolean variables $X_{ij1}, \dots, X_{ijn-1}$
- $X_{ij} = l$ for $l = 1, \dots n-1$: $\overline{X_{ij1}} \wedge \overline{X_{ij2}} \wedge \dots \wedge \overline{X_{ijl-1}} \wedge X_{ijl}$,
- $X_{ij} = n$: $\overline{X_{ij1}} \wedge \overline{X_{ij2}} \wedge \dots \wedge \overline{X_{ijn-1}}$.
- Parameters: $P(X_{ij1}) = P(X_{ij} = 1) \dots P(X_{ijl}) = \frac{P(X_{ij}=l)}{\prod_{m=1}^{l-1}(1-P(X_{ijm}))}$.



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

Monte Carlo

- 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 $MC(C_r)$
  $MC(C_r, 1) = H_1 \leftarrow L_1, \ldots, L_m, sample\_head(n, r, VC, NH), NH = 1.$
  ...
  $MC(C_r, n) = H_n \leftarrow L_1, \ldots, L_m, sample\_head(n, r, VC, NH), NH = n.$
- Sample truth value of query `Q`:

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

Dipartimento di Matematica e Informatica

Inference in DISPONTE

- The probability of a query Q can be computed according to the distribution semantics by first finding the explanations for Q in the knowledge base
- Explanation: subset of axioms of the KB that is sufficient for entailing Q
- All the explanations for Q must be found, corresponding to all ways of proving Q

Dipartimento di Matematica e Informatica

Inference in DISPONTE

- Probability of $Q \rightarrow$ probability of the DNF formula

$$F(Q) = \bigvee_{e \in E_Q} \left(\bigwedge_{F_i \in e} X_i \right)$$

where E_Q is the set of explanations and X_i is a Boolean random variable associated to axiom F_i

- Binary Decision Diagrams for efficiently computing the probability of the DNF formula

Example

$E_1 = 0.4 :: \mathit{fluffy} : \mathit{Cat}$
$E_2 = 0.3 :: \mathit{tom} : \mathit{Cat}$
$E_3 = 0.6 :: \mathit{Cat} \sqsubseteq \mathit{Pet}$
$\exists \mathit{hasAnimal}.\mathit{Pet} \sqsubseteq \mathit{NatureLover}$
$(\mathit{kevin}, \mathit{fluffy}) : \mathit{hasAnimal}$
$(\mathit{kevin}, \mathit{tom}) : \mathit{hasAnimal}$

- $Q = \mathit{kevin} : \mathit{NatureLover}$ has two explanations:

$$\{\ (E_1), (E_3)\ \}$$
$$\{\ (E_2), (E_3)\ \}$$

- $P(Q) = 0.4 \times 0.6 \times (1 - 0.3) + 0.3 \times 0.6 = 0.348$

BUNDLE

- Binary decision diagrams for Uncertain reasoNing on Description Logic thEories [Riguzzi et al. SWJ15]
- BUNDLE performs inference over DISPONTE knowledge bases.
- It exploits an underlying ontology reasoner able to return all explanations for a query, such as **Pellet** [Sirin et al, WS 2007]
- Then DNF formula built and converted to BDDs for computing the probability

Dipartimento di Matematica e Informatica

TRILL

- Tableau Reasoner for descrIption Logics in proLog
- TRILL implements the tableau algorithm using Prolog
- It resolves the axiom pinpointing problem in which we are interested in the set of explanations that entail a query
- It returns the set of the explanations
- It can build BDDs encoding the set of explanations and return the probability

TRILL

- Available online at `http://trill-sw.eu/`
- Pets example
  `http://trill-sw.eu/example/trill/peoplePets.pl`

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

- 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

Parameter Learning

- [Thon et al. ECML 2008] proposed an adaptation of EM for CPT-L, a simplified version of LPADs
- The algorithm computes the counts efficiently by repeatedly traversing the BDDs representing the explanations
- [Ishihata et al. ILP 2008] independently proposed a similar algorithm
- LFI-PROBLOG [Gutamnn et al. ECML 2011]: EM for ProbLog
- EMBLEM [Riguzzi & Bellodi IDA 2013] adapts [Ishihata et al. ILP 2008] to LPADs

EMBLEM

- 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

EDGE

- Em over bDds for description loGics paramEter learning
- EDGE is inspired to EMBLEM [Bellodi and Riguzzi, IDA 2013]
- Takes as input a DL theory and a number of examples that represent queries.
- The queries are concept assertions and are divided into:
  1. positive examples;
  2. negative examples.
- EDGE computes the explanations of each example using BUNDLE, that builds the corresponding BDD.
  - For negative examples, EDGE computes the explanations of the query, builds the BDD and then negates it.

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 EMBLEM [Riguzzi & Bellodi TPLP 2015]
  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 NGC 1995]

Mode Declarations

- Syntax

```
modeh(RecallNumber,PredicateMode).
modeb(RecallNumber,PredicateMode).
```

- `RecallNumber` can be a number or *. Usually *. Maximum number of answers to queries to include in the bottom clause
- `PredicateMode`: template of the form:

```
p(ModeType, ModeType,...)
```

Dipartimento di Matematica e Informatica

Mode Declarations

- `ModeType` can be:
  - Simple:
    - `+T` input variables of type `T`;
    - `-T` output variables of type `T`; or
    - `#T`, `-#T` constants of type `T`.
  - Structured: of the form `f(..)` where `f` is a function symbol and every argument can be either simple or structured.

Dipartimento di Matematica e Informatica

Mode Declarations

- Some examples:

```
modeb(1,mem(+number,+list)).
modeb(1,dec(+integer,-integer)).
modeb(1,mult(+integer,+integer,-integer)).
modeb(1,plus(+integer,+integer,-integer)).
modeb(1,(+integer)=(#integer)).
modeb(*,has_car(+train,-car))
modeb(1,mem(+number,[+number|+list])).
```

Dipartimento di Matematica e Informatica

Bottom Clause $\bot$

- Most specific clause covering an example e
- Form: $e \leftarrow B$
- B: set of ground literals that are true regarding the example e
- B obtained by considering the constants in e and querying the predicates of the background for true atoms regarding these constants
- A map from types to lists of constants is kept, it is enlarged with constants in the answers to the queries and the procedure is iterated a user-defined number of times
- Values for output arguments are used as input arguments for other predicates

Bottom Clause $\perp$

- Initialize to empty a map *m* from types to lists of values
- Pick a $modeh(r, s)$, an example *e* matching *s*, add to $m(T)$ the values of $+T$ arguments in *e*
- For $i = 1$ to *d*
  - For each $modeb(r, s)$

Dipartimento
di Matematica
e Informatica

Bottom Clause $\perp$

- For each possible way of building a query q from s by replacing $+T$ and $\#T$ arguments with constants from $m(T)$ and all other arguments with variables
  - Find all possible answers for q and put them in a list L
  - $L' := r$ elements sampled from L
  - For each $l \in L'$, add the values in l corresponding to $-T$ or $-\#T$ to $m(T)$

Dipartimento di Matematica e Informatica

Bottom Clause $\bot$

- Example:

$e = father(john, mary)$
$B = \{parent(john, mary), parent(david, steve),$
$parent(kathy, mary), female(kathy), male(john), male(david)\}$
$modeh(father(+person, +person)).$
$modeb(parent(+person, -person)).$
$modeb(parent(-\#person, +person)).$
$modeb(male(+person)).$ $modeb(female(\#person)).$
$e \leftarrow B = father(john, mary) \leftarrow parent(john, mary), male(john),$
$parent(kathy, mary), female(kathy).$

Bottom Clause $\bot$

- The resulting ground clause $\bot$ is then processed by replacing each term in a + or - placemarker with a variable
- An input variable (+T) must appear as an output variable with the same type in a previous literal and a constant (#T or -#T) is not replaced by a variable.

$\bot = father(X, Y) \leftarrow$
$parent(X, Y), male(X), parent(kathy, Y), female(kathy).$

Dipartimento di Matematica e Informatica

SLIPCOVER

- The initial beam associated with predicate P/Ar will contain the clause with the empty body $h : 0.5.$ for each bottom clause $h :- b_1, \ldots, b_m$
- In each iteration of the cycle over predicates, it performs a beam search in the space of clauses for the predicate.
- The beam contains couples $(Cl, LIterals)$ where $Literals = \{b_1, \ldots, b_m\}$
- For each clause Cl of the form $Head :- Body$, the refinements are computed by adding a literal from $Literals$ to the body.

SLIPCOVER

- The tuple (Cl', $Literals'$) indicates a refined clause Cl' together with the new set $Literals'$
- EMBLEM is then executed for a theory composed of the single refined clause.
- LL is used as the score of the updated clause $(Cl'', Literals')$.
- $(Cl'', Literals')$ is then inserted into a list of promising clauses.
- Two lists are used, *TC* for target predicates and *BC* for background predicates.
- These lists ave a maximum size

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 clauses in *BC* to the theory and performs parameter learning on the resulting theory.

Dipartimento di Matematica e Informatica

Experiments - Area Under the PR Curve

System	HIV	UW-CSE	Mondial
SLIPCOVER	0.82 ± 0.05	0.11 ± 0.08	0.86 ± 0.07
SLIPCASE	0.78 ± 0.05	0.03 ± 0.01	0.65 ± 0.06
LSM	0.37 ± 0.03	0.07 ± 0.02	-
ALEPH++	-	0.05 ± 0.01	0.87 ± 0.07
RDN-B	0.28 ± 0.06	0.28 ± 0.06	0.77 ± 0.07
MLN-BT	0.29 ± 0.04	0.18 ± 0.07	0.74 ± 0.10
MLN-BC	0.51 ± 0.04	0.06 ± 0.01	0.59 ± 0.09
BUSL	0.38 ± 0.03	0.01 ± 0.01	-

Dipartimento di Matematica e Informatica

Experiments - Area Under the PR Curve

System	Carcinogenesis	Mutagenesis	Hepatitis
SLIPCOVER	0.60	0.95 ± 0.01	0.80 ± 0.01
SLIPCASE	0.63	0.92 ± 0.08	0.71 ± 0.05
LSM	-	-	0.53 ± 0.04
ALEPH++	0.74	0.95 ± 0.01	-
RDN-B	0.55	0.97 ± 0.03	0.88 ± 0.01
MLN-BT	0.50	0.92 ± 0.09	0.78 ± 0.02
MLN-BC	0.62	0.69 ± 0.20	0.79 ± 0.02
BUSL	-	-	0.51 ± 0.03

Dipartimento di Matematica e Informatica

Bongard Problems

- Introduced by the Russian scientist M. Bongard
- Pictures, some positive and some negative
- Problem: discriminate between the two classes.
- The pictures contain shapes with different properties, such as small, large, pointing down, . . . and different relationships between them, such as inside, above, . . .

Dipartimento di Matematica e Informatica

Input File

http://cplint.eu/e/bongard.pl

```
:- use_module(library(slipcover)).
:- if(current_predicate(use_rendering/1)).
:- use_rendering(c3).
:- use_rendering(lpad).
:- endif.
:-sc.
:- set_sc(megaex_bottom,20).
:- set_sc(max_iter,3).
:- set_sc(max_iter_structure,10).
:- set_sc(maxdepth_var,4).
:- set_sc(verbosity,1).
```

See `http://cplint.eu/help/help-cplint.html` for a list of options

Input File

Theory for parameter learning and background

```
bg([]).
in([
(pos:0.5 :-
  circle(A),
  in(B,A)),
(pos:0.5 :-
  circle(A),
  triangle(B))]).
```

Dipartimento di Matematica e Informatica

Input File

Data: two formats, models

```
begin(model(2)).
pos.
triangle(o5).
config(o5,up).
square(o4).
in(o4,o5).
circle(o3).
triangle(o2).
config(o2,up).
in(o2,o3).
triangle(o1).
config(o1,up).
end(model(2)).

begin(model(3)).
neg(pos).
circle(o4).
circle(o3).
in(o3,o4).
....
```

Dipartimento di Matematica e Informatica

Input File

Data: two formats, keys (internal representation)

```
pos(2).
triangle(2,o5).
config(2,o5,up).
square(2,o4).
in(2,o4,o5).
circle(2,o3).
triangle(2,o2).
config(2,o2,up).
in(2,o2,o3).
triangle(2,o1).
config(2,o1,up).

neg(pos(3)).
circle(3,o4).
circle(3,o3).
in(3,o3,o4).
square(3,o2).
circle(3,o1).
in(3,o1,o2).
....
```

Dipartimento di Matematica e Informatica

Input File

- Folds
- Target predicates: `output(<predicate>)`
- Input predicates are those whose atoms you are not interested in predicting

  ```
  input_cw(<predicate>/<arity>).
  ```

  True atoms are those in the interpretations and those derivable from them using the background knowledge
- Open world input predicates are declared with

  ```
  input(<predicate>/<arity>).
  ```

  the facts in the interpretations, the background clauses and the clauses of the input program are used to derive atoms

Input File

```
fold(train,[2,3,5,...]).
fold(test,[490,491,494,...]).
output(pos/0).
input_cw(triangle/1).
input_cw(square/1).
input_cw(circle/1).
input_cw(in/2).
input_cw(config/2).
```

Dipartimento
di Matematica
e Informatica

Input File

- Language bias
- *determination*($p/n, q/m$): atoms for q/m can appear in the body of rules for p/n

```
determination(pos/0,triangle/1).
determination(pos/0,square/1).
determination(pos/0,circle/1).
determination(pos/0,in/2).
determination(pos/0,config/2).
modeh(*,pos).
modeb(*,triangle(-obj)).
modeb(*,square(-obj)).
modeb(*,circle(-obj)).
modeb(*,in(+obj,-obj)).
modeb(*,in(-obj,+obj)).
modeb(*,config(+obj,-#dir)).
```

Input File

Search bias

```
lookahead(logp(B),[(B=_C)]).
```

Bongard Problems

- Parameter learning

```
induce_par([train],P),
  test(P,[test],LL,AUCROC,ROC,AUCPR,PR).
```

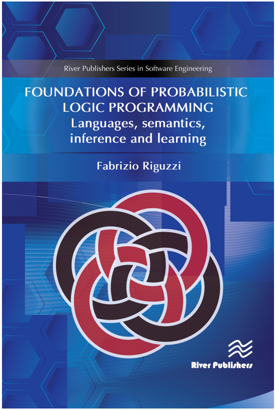
- Structure learning

```
induce([train],P),
  test(P,[test],LL,AUCROC,ROC,AUCPR,PR).
```

Dipartimento di Matematica e Informatica

Conclusions

- Exact inference
- Approximate inference
- Parameter learning
- Structure learning
- Research directions:
  - Structure learning search strategies
  - Learning hybrid programs
  - Learning restricted and cheaper languages



Dipartimento di Matematica e Informatica

Resources

- Online course on cplint
  - Moodle `https://edu.swi-prolog.org/`
  - Videos of lectures `https://www.youtube.com/playlist?list=PLJPXEH0boeND0UGWJxBRWs7qzzKpC-FkN`
- ACAI summer school on Statistical Relational AI `http://acai2018.unife.it/`
- Videos of lectures `https://www.youtube.com/playlist?list=PLJPXEH0boeNDWTNwWTWnVffXi5XwAj1mb`
- Videos of lecture Probabilistic Inductive Logic Programming
  - Part 1 `https://youtu.be/mLdPGSlgNxU`
  - Part 2 `https://youtu.be/DRlOft0Y_Ng`
- cplint in Playing with Prolog `https://www.youtube.com/playlist?list=PLJPXEH0boeNAik6QnfvGlAGRQxFY_LCE3`

Dipartimento di Matematica e Informatica

THANKS FOR LISTENING

AND

ANY QUESTIONS ?

Dipartimento
di Matematica
e Informatica

References

- Bellodi, E. and Riguzzi, F. (2012). Learning the structure of probabilistic logic programs. In Inductive Logic Programming 21st International Conference, ILP 2011, London, UK, July 31 - August 3, 2011. Revised Papers, volume 7207 of LNCS, pages 61-75, Heidelberg, Germany. Springer.
- Bellodi, E. and Riguzzi, F. (2013). Expectation Maximization over binary decision diagrams for probabilistic logic programs. Intelligent Data Analysis, 17(2).
- Gutmann, B., Thon, I., and Raedt, L. D. (2011). Learning the parameters of probabilistic logic programs from interpretations. In European Conference on Machine Learning and Knowledge Discovery in Databases, volume 6911 of LNCS, pages 581-596. Springer.

Dipartimento di Matematica e Informatica

References

- Muggleton, S. (1995). Inverse entailment and progol. New Generation Comput., 13(3&4):245-286.
- Riguzzi, F. (2007). A top down interpreter for LPAD and CP-logic. In Congress of the Italian Association for Artificial Intelligence, number 4733 in LNAI, pages 109-120. Springer.
- Riguzzi, F. (2009). Extended semantics and inference for the Independent Choice Logic. Logic Journal of the IGPL.
- Riguzzi, F. and Swift, T. (2010). Tabling and Answer Subsumption for Reasoning on Logic Programs with Annotated Disjunctions. In Hermenegildo, M. and Schaub, T., editors, Technical Communications of the 26th Int'l. Conference on Logic Programming (ICLP10), volume 7 of Leibniz International Proceedings in Informatics (LIPIcs), pages 162-171, Dagstuhl, Germany. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Dipartimento di Matematica e Informatica

References

- Sato, T. (1995). A statistical learning method for logic programs with distribution semantics. In International Conference on Logic Programming, pages 715-729.
- Thon, I., Landwehr, N., and Raedt, L. D. (2008). A simple model for sequences of relational state descriptions. In Daelemans, W., Goethals, B., and Morik, K., editors, Machine Learning and Knowledge Discovery in Databases, European Conference, ECML/PKDD 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings, Part II, volume 5212 of Lecture Notes in Computer Science, pages 506-521. Springer.

Dipartimento di Matematica e Informatica