EM over Binary Decision Diagrams for Probabilistic Logic Programs

Elena Bellodi    Fabrizio Riguzzi

ENDIF – University of Ferrara, Italy
elena.bellodi@unife.it, fabrizio.riguzzi@unife.it
1. Probabilistic Logic Languages
2. Inference with Decision Diagrams
3. Weight Learning for LPADs
4. EM over BDDs
5. Experiments and results
6. Conclusions and future works
7. References
Probabilistic Logic Programming

- Logic + Probability: useful to model domains with *complex* and *uncertain* relationships among entities
- Many approaches proposed in: *Logic Programming, Uncertainty in AI, Machine Learning, Databases*

**Logic Programming**: Distribution Semantics [Sato, 1995]
- Independent Choice Logic, PRISM, ProbLog, Logic Programs with Annotated Disjunctions (LPADs) [Vennekens et al., 2004],...
- They define a probability distribution over normal logic programs (possible worlds)
- They differ in the definition of the probability distribution
- The distribution is extended to a joint distribution over worlds and queries
- The probability of a query is obtained from this distribution by marginalization
Example: development of an epidemic or pandemic, if somebody has the flu and the climate is cold.

\[ C_1 = \text{epidemic} : 0.6; \text{pandemic} : 0.3; \text{null} : 0.1 : \neg \text{flu}(X), \text{cold}. \]
\[ C_2 = \text{cold} : 0.7; \text{null} : 0.3. \]
\[ C_3 = \text{flu}(\text{david}). \]
\[ C_4 = \text{flu}(\text{robert}). \]

Worlds obtained by selecting only one atom from the head of every grounding of each rule
Inference

- **Explanation**: set of probabilistic choices that ensure the entailment of the goal

- **Covering set of explanations**: every world where the query is true is consistent with at least one explanation

- A covering set of explanations for :- epidemic. is \( \{ \kappa_1, \kappa_2 \} \)
  
  \( \kappa_1 = \{(C_1, \theta_1 = \{X/david\}, 1), (C_2, \{\}, 1)\} \)
  
  \( \kappa_2 = \{(C_1, \theta_2 = \{X/robert\}, 1), (C_2, \{\}, 1)\} \)

- Explanations are not mutually exclusive

- From a covering set of explanations the probability of the query \( Q \) is computed by means of Decision Diagrams
Multivalued Decision Diagrams (MDDs) represent a Boolean function $f(X)$ on a set of multivalued variables $X_{ij} \rightarrow$ ground clause $C_i \theta_j$, with domain $1,...,|head(C_i)|$

In a MDD a path to a 1-leaf corresponds to an explanation for $Q$

The various paths are mutually exclusive

$$f(X) = (X_{11} = 1 \land X_{21} = 1) \lor (X_{12} = 1 \land X_{21} = 1)$$
Binary Decision Diagrams (BDD)

- MDDs can be converted into Binary Decision Diagrams with Boolean variables.
- Multivalued variable $X_{ij}$ with $n_i$ values $\rightarrow n_i - 1$ Boolean variables $X_{ij1}, \ldots, X_{ijn_i-1}$.
- From $f(X) = (X_{11} = 1 \land X_{21} = 1) \lor (X_{12} = 1 \land X_{21} = 1)$ to $f(X) = ((X_{111} \land \overline{X_{112}}) \land X_{211}) \lor ((X_{121} \land X_{122}) \land X_{211})$. 

![Decision Diagram Example]

- $X_{111}$
- $X_{121}$
- $X_{211}$
- $n_1$
- $n_2$
- $n_3$
- 1
- 0
Weight Learning for LPADs

- **Problem**: model of the domain known vs weights (numeric parameters) unknown

- **Weight learning**: inference of weights from data

- **Given**
  - a LPAD: a probabilistic logical model with unknown probabilities
  - data: a set of interpretations
  - **Find** the values of the probabilities that maximize the probability of the data given the model

- **Expectation Maximization (EM) algorithm**
  - iterative method for problems with incomplete data
  - *Expectation* step: estimates missing data given observed data + current estimate of parameters
  - *Maximization* step: computes the parameters using estimates of E step
EM over BDDs proposed in [Ishihata et al., 2008]

Input: a 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 disjunction of explanations for each query Q

EM algorithm directly over the BDDs

Missing data: the number of times that i-th head atom has been selected from groundings of the clauses used in the proof of the queries
EM Algorithm

- **Expectation step** (synthesis)

1. Computes $P(X_{ijk} = x, Q)$ and $P(Q)$
2. Computes expected counts $E[c_{ikx}] = \frac{\sum_{j \in g(i)} P(X_{ijk} = x, Q)}{P(Q)}$
   for all rules $C_i$ and $k = 1, ..., n_i - 1$, where $c_{ikx}$ is the number of times a binary variable $X_{ijk}$ takes value $x \in \{0, 1\}$, and for all values of $j \in g(i) = \{j | \theta_j \text{ is a substitution grounding } C_i\}$

- **Maximization step**

- Updates parameters $\pi_{ik}$ representing $P(X_{ijk} = 1)$
  
  $\pi_{ik} = E[c_{ik1} | Q] / (E[c_{ik0} | Q] + E[c_{ik1} | Q])$
Expectation Computation

\[ P(X_{ijk} = x, Q) = \sum_{n \in N(Q), v(n) = x_{ijk}} F(n) B(\text{child}_x(n)) \pi_{ikx} = \sum_{n \in N(Q), v(n) = x_{ijk}} e^x(n) \]

- \( \pi_{ikx} \) is \( \pi_{ik} \) if \( x = 1 \) and \((1 - \pi_{ik})\) if \( x = 0 \)
- \( F(n) \) is the forward probability, the probability mass of the paths from the root to \( n \)
- \( B(n) \) is the backward probability, the probability mass of paths from \( n \) to the 1-leaf
- \( e^x(n) \) is the probability mass of paths from the root to the 1 leaf passing through the \( x \) branch of \( n \)
Computation of the forward probability

1: procedure GET_FORWARD(root)
2: \( F(root) = 1 \) \( F(n) = 0 \) for all nodes \( \triangleright \) BDD traversed from root to leaves
3: for \( l = 1 \) to levels do \( \triangleright \) BDD levels
4:   for all node \( \in \) Nodes(\( l \)) do \( \triangleright \) Nodes of one level
5:     Let \( X_{ijk} \) be \( v(node) \), the variable associated to node
6:     if \( \text{child}_0(node) \) is not terminal then \( \triangleright \) node’s child connected by 0-branch
7:         \( F(\text{child}_0(node)) = F(\text{child}_0(node)) + F(node) \cdot (1 - \pi_{ik}) \) \( \triangleright \pi_{ik}: \text{probability} \)
8:         Add \( \text{child}_0(node) \) to Nodes(\( \text{level}(\text{child}_0(node)) \))
9:     end if
10:    if \( \text{child}_1(node) \) is not terminal then
11:        \( F(\text{child}_1(node)) = F(\text{child}_1(node)) + F(node) \cdot \pi_{ik} \)
12:        Add \( \text{child}_1(node) \) to Nodes(\( \text{level}(\text{child}_1(node)) \))
13:    end if
14: end for
15: end for
16: end procedure

\( \circ \) For all nodes of a level the forward probabilities of their children are computed, by using probabilities \( \pi_{ik} \) associated to the outgoing edges
Computation of the backward probability

function GET_BACKWARD(node)
    if node is a terminal then
        return value(node)
    else
        Let $X_{ijk}$ be $v(node)$
        $B(child_0(node)) = \text{GET_BACKWARD}(child_0(node))$
        $B(child_1(node)) = \text{GET_BACKWARD}(child_1(node))$
        $e^0(node) = F(node) \cdot B(child_0(node)) \cdot (1 - \pi_{ik})$
        $e^1(node) = F(node) \cdot B(child_1(node)) \cdot \pi_{ik}$
        $\eta^0(i, k) = \eta^0_t(i, k) + e^0(node)$
        $\eta^1(i, k) = \eta^1_t(i, k) + e^1(node)$
        return $B(child_0(node)) \cdot (1 - \pi_{ik}) + B(child_1(node)) \cdot \pi_{ik}$
    end if
end function

at the end of all recursive calls, the function returns $B(root) =$ probability of the query $P(Q)$
Experiments and results

Experiments - settings

- **EMBLEM** is implemented in Yap Prolog

- **Comparison with other systems**
  - for learning and inference under the distribution semantics:
    - RIB [Riguzzi and di Mauro, 2011]
    - CEM [Riguzzi, 2007]
    - LeProblog [De Raedt et al., 2007]
  - for learning and inference in Markov Logic Networks: **Alchemy**

- Datasets composed of 5 mega-interpretations → Five-fold cross validation

- **Performance evaluation**
  - Area Under the PR (Precision-Recall) Curve
Experiments and results

Experiments - datasets

- **IMDB** - the Internet Movie DataBase: movies, actors, directors.

Input LPADs

- **target predicate** `sameperson(per1, per2)`

  \[
  \text{sameperson}(X, Y) : p : \neg \text{movie}(M, X), \text{movie}(M, Y).
  \]
  \[
  \text{sameperson}(X, Y) : p : \neg \text{actor}(X), \text{actor}(Y), \text{workedunder}(X, Z), \text{workedunder}(Y, Z).
  \]
  \[
  \text{sameperson}(X, Y) : p : \neg \text{gender}(X, Z), \text{gender}(Y, Z).
  \]
  \[
  \text{sameperson}(X, Y) : p : \neg \text{director}(X), \text{director}(Y), \text{genre}(X, Z), \text{genre}(Y, Z).
  \]

- **target predicate** `samemovie(mov1, mov2)`

  \[
  \text{samemovie}(X, Y) : p : \neg \text{movie}(X, M), \text{movie}(Y, M), \text{actor}(M).
  \]
  \[
  \text{samemovie}(X, Y) : p : \neg \text{movie}(X, M), \text{movie}(Y, M), \text{director}(M).
  \]
  \[
  \text{samemovie}(X, Y) : p : \neg \text{movie}(X, A), \text{movie}(Y, B), \text{actor}(A), \text{director}(B), \text{workedunder}(A, B).
  \]
  \[
  \text{samemovie}(X, Y) : p : \neg \text{movie}(X, A), \text{movie}(Y, B), \text{director}(A), \text{director}(B), \text{genre}(A, G), \text{genre}(B, G).
  \]

### Average of the AUCPR

<table>
<thead>
<tr>
<th>Dataset</th>
<th>EMBLEM</th>
<th>RIB</th>
<th>LeProblog</th>
<th>CEM</th>
<th>Alchemy</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMDB-SP</td>
<td>0.202</td>
<td>0.199</td>
<td>0.096</td>
<td>0.202</td>
<td>0.107</td>
</tr>
<tr>
<td>IMDB-SM</td>
<td>1.000</td>
<td>memory error</td>
<td>0.933</td>
<td>0.537</td>
<td>0.820</td>
</tr>
</tbody>
</table>

Bellodi, Riguzzi (University of Ferrara)
Experiments - datasets

**CORA**: citations to computer science research papers

[Singla and Domingos, 2005]

**target predicate**: $\text{samebib}(\text{cit}_1, \text{cit}_2)$, to determine which citations are referring to the same paper

**Input LPADs**

- 559 clauses
  
  $\text{samebib}(B, C) : p : \neg \text{author}(B, D), \text{author}(C, E), \text{sameauthor}(D, E)$.
  
  $\text{samebib}(B, C) : p : \neg \text{title}(B, D), \text{title}(C, E), \text{sametitle}(D, E)$.
  
  $\text{samebib}(B, C) : p : \neg \text{venue}(B, D), \text{venue}(C, E), \text{samevenue}(D, E)$.
  
  $\text{samevenue}(B, C) : p : \neg \text{haswordvenue}(B, W^*), \text{haswordvenue}(C, W^*). *W$ instantiated to all words
  
  $\text{sametitle}(B, C) : p : \neg \text{haswordtitle}(B, W^*), \text{haswordtitle}(C, W^*)$.
  
  $\text{sameauthor}(B, C) : p : \neg \text{haswordauthor}(B, W^*), \text{haswordauthor}(C, W^*)$.

- + 4 transitive rules (CoraT)
  
  $\text{samebib} / \text{title} / \text{author} / \text{venue}(A, B) : p : \neg \text{samebib} / \text{title} / \text{author} / \text{venue}(A, C)$,
  
  $\text{samebib} / \text{title} / \text{author} / \text{venue}(C, B)$.

### Average of the AUCPR

<table>
<thead>
<tr>
<th>Dataset</th>
<th>EMBLEMEM</th>
<th>RIB</th>
<th>LeProblog</th>
<th>CEM</th>
<th>Alchemy</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORA</td>
<td>0.995</td>
<td>0.939</td>
<td>0.905</td>
<td>0.995</td>
<td>0.469</td>
</tr>
<tr>
<td>CORAT</td>
<td>0.991</td>
<td>not appl.</td>
<td>0.970</td>
<td>memory error</td>
<td>memory error</td>
</tr>
</tbody>
</table>

Bellodi, Riguzzi (University of Ferrara)
Experiments - datasets

- **UWCSE**: information about the Computer Science department of the University of Washington [Kok and Domingos, 2010]
- **target predicate**: `advisedBy/2`, a person is advised by another person
- **Input LPAD**: 86 clauses, such as

  \[
  \text{advisedby}(S, P) : p : -\text{courselevel}(C, \text{level} \_500), \text{taughtby}(C, P, Q), \text{ta}(C, S, Q). \\
  \text{tempadvisedby}(S, P) : p : -\text{courselevel}(C, \text{level} \_500), \text{taughtby}(C, P, Q), \text{ta}(C, S, Q). \\
  \text{professor}(P) : p : -\text{courselevel}(C, \text{level} \_500), \text{taughtby}(C, P, Q).
  \]

<table>
<thead>
<tr>
<th>Average of the AUCPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMBLEM</td>
</tr>
<tr>
<td>0.883</td>
</tr>
</tbody>
</table>
Conclusions and future works

**EMBLEM** can be applied to all languages based on the distribution semantics, since there are transformations with linear complexity that can convert a program in one language into the others.

- Able to solve greater problems, where other algorithms do not terminate.
- Higher PR areas with same learning time as the fastest other algorithm.
- Higher PR areas with longer learning time.

*In progress*: structure learning of LPADs (clauses+parameters).


