Principles of Managing Uncertain Data

Lecture 12: More on Probabilistic Databases
Table of Contents

1 Introduction
2 Markov Logic Networks
3 Probabilistic XML
Many thanks to Dan Suciu for advising on these slides!
Table of Contents

1 Introduction

2 Markov Logic Networks

3 Probabilistic XML
So far, we have learned about representations of probabilistic databases

Focusing on tuple-independent databases (TID), we studied query evaluation in depth

This lecture: two additional notions of probabilistic databases:

- Markov Logic Networks
- Probabilistic XML
Table of Contents

1 Introduction

2 Markov Logic Networks

3 Probabilistic XML
Running Example

We wish to guess the missing values; *but no reason to prefer one guess to another...* yet
## Soft Inference Rules

<table>
<thead>
<tr>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p2</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Bob</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Chloe</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
<tr>
<td>p</td>
<td>SoccerFan(x) ∧ Follows(y, x) → SoccerFan(y)</td>
<td></td>
</tr>
<tr>
<td>p</td>
<td>OperaFan(x) ∧ Follows(y, x) → OperaFan(y)</td>
<td></td>
</tr>
<tr>
<td>t</td>
<td>Soccer(x) ↔ ¬OperaFan(x)</td>
<td></td>
</tr>
</tbody>
</table>
### Soft Inference Rules

<table>
<thead>
<tr>
<th></th>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>p1</strong></td>
<td>p</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Bob</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Chloe</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
<tr>
<td><strong>p2</strong></td>
<td>t</td>
<td>?</td>
<td>f</td>
</tr>
<tr>
<td><strong>4:</strong></td>
<td>((\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y))</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2:</strong></td>
<td>((\text{OperaFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{OperaFan}(y))</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8:</strong></td>
<td>\text{Soccer}(x) \leftrightarrow \neg(\text{OperaFan}(x))</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Interpreting Soft Rules

- The semantics of such a database with soft rules is a probability space over the completions of the unknown values.

  - *How softness and weighting translate into probabilities?*
    - Various semantics in the literature, e.g., Probabilistic Datalog [Fuh00], ProbLog [RKT07], Probabilistic Soft Logic [BMG10], Probabilistic-Programming Datalog [BtCK+17]

- We will look at one such a translation: *Markov Logic Network (MLN)* [RD06]
We have a sequence $\mathbf{z} = z_1, \ldots, z_n$ of (correlated) random variables, each $z_i$ taking values from a domain $\text{dom}(z_i)$.

We consider the representation of a probability space over all possible assignments.
Factors

- A factor over \( z = z_1, \ldots, z_n \) is a function

\[
\phi : \text{dom}(z_1) \times \cdots \times \text{dom}(z_n) \to [0, \infty)
\]

- We are typically interested in situations where each factor looks at only a small subset of the variables, e.g.:

\[
\phi(a) = \begin{cases} 
2 & \text{if } a_1 = 0 \\
5 & \text{if } a_1 = 1 \text{ and } a_7 = 0 \\
15 & \text{if } a_1 = 1 \text{ and } a_7 = 1 
\end{cases}
\]
A factor graph is a representation of a probability space over the assignments to \( z \).

Formally, a **factor graph** for \( z \) is a sequence \( F = \phi_1, \ldots, \phi_m \) of factors over \( z \).

**Semantics:**

\[
\text{Pr}(z = a) \overset{\text{def}}{=} \frac{1}{Z} \prod_{\phi_i \in F} \phi_i(a)
\]

\( Z \) is a **normalization term**:

\[
Z \overset{\text{def}}{=} \sum_a \prod_{\phi_i \in F} \phi_i(a)
\]
Visually, it is convenient and conventional to represent a factor graph by a bipartite graph with:

- A node $v_f$ for each factor $f$
- A node $u_{z_i}$ for every random variable $z_i$ in $z$
- An edge between $v_f$ and $u_{z_i}$ whenever $z_i$ affects $f$
Example

![Factor Graph Example]

- f₁ is connected to AnnaSoccerFan.
- f₂ is connected to AnnaOperaFan.
- f₃ is connected to AnnaOperaFan indirectly via f₁.
- f₄ is connected to BobSoccerFan.
- f₅ is connected to BobOperaFan.
- f₆ is connected to BobOperaFan indirectly via f₂.
- f₇ is connected to ChloeSoccerFan.
- f₈ is connected to ChloeOperaFan.
- f₉ is connected to ChloeOperaFan indirectly via f₂ and f₃.
Example: Naïve Bayes Classifier

\[
\Pr(y = c, x = a) = \Pr(y = c) \cdot \prod_{i=1}^{n} \Pr(x_i = a_i \mid y = c)
\]

\[
= \phi(c) \cdot \prod_{i=1}^{n} \phi_i(a_i, c)
\]
Example: Logistic Regression

\[
\Pr(y = c, \mathbf{x} = \mathbf{a}) = \frac{e^{\beta_{c,0} + \sum \beta_{c,i} \cdot a_i}}{Z}
\]
Example: Hidden Markov Model

\[
\text{Pr}(y = c, x = a) = \text{Pr}(y_1 = c_1) \cdot \text{Pr}(x_1 = a_1 \mid y_1 = c_1) \\
\times \text{Pr}(y_2 = c_2 \mid y_1 = c_1) \cdot \text{Pr}(x_2 = a_2 \mid y_2 = c_2) \\
\times \ldots \\
\times \text{Pr}(y_n = c_n \mid y_{n-1} = c_{n-1}) \cdot \text{Pr}(x_n = a_n \mid y_n = c_n)
\]
A Markov Logic Network (MLN) is a formalism for compactly encoding factor graphs using logic

- facts $\Rightarrow$ Boolean random variables
- rules $\Rightarrow$ factors
Some Applications of MLNs

- Entity resolution [SD06]
- Information extraction (extract structured data from text) [NRDS11]
- Robotics (scene analysis) [WD08]
- Social-network analysis
- Bioinformatics (protein interaction)
We have a relational signature $\mathcal{R}$ and a finite domain $C$ of attribute values

- For example, $\mathcal{R} = \{\text{Follows}/2, \text{SoccerFan}/1, \text{OperaFan}/1\}$ and $C = \{\text{Anna}, \text{Bob}, \text{Chloe}\}$
- For simplicity, same domain for all attributes
A ground fact over $\mathcal{R}$ is a fact over $\mathcal{R}$ with values from $C$,
- e.g., SoccerFan(Anna), Follows(Bob, Anna)

A Markov Logic Network (MLN) represents a probability space over all possible sets of ground facts.

We view a ground fact $f$ as a random Boolean variable $x_f$
- $x_f = \text{true}$ means that the fact $f$ is true (e.g., Anna is indeed a soccer fan)

An MLN is then a probability space over the $x_f$, assigning random true/false to each variable.
An MLN over the signature \( \mathcal{R} \) and domain \( \mathcal{C} \) is a sequence \( \langle w_1 : r_1, \ldots, w_k : r_k \rangle \) of weighted rules:

- Each \( r_i \) is a propositional formula with free variables.
- Each \( w_i \) is a nonnegative number.
- Zeros are important for encoding hard constraints (must hold in every world).

A grounding of a rule \( r \) is obtained by replacing the free variables with constants; for example:

\[
(\text{OperaFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{OperaFan}(y)
\]

\[
\downarrow
\]

\[
(\text{OperaFan}(\text{Anna}) \land \text{Follows}(\text{Bob, Anna})) \rightarrow \text{OperaFan}(\text{Bob})
\]
The MLN $\langle w_1 : r_1, \ldots, w_k : r_k \rangle$ represents a factor graph over the $x_f$

A possible assignment $a$ to the sequence $x$ of variables represents an instance over $\mathcal{R}$; we denote it by $I_a$

$\cdot$ $I_a$ consists of all the facts $f$ such that $a_f = \text{true}$

Each grounding $g$ of $r_i$ provides a factor, denoted $\phi_{g,r_i}$

$$\phi_{g,r_i}(a) = \begin{cases} w_i & \text{if } g(r_i) \text{ is true in the assignment } I_a; \\ 1 & \text{otherwise.} \end{cases}$$

What are the random variables that connect to $\phi_{g,r_i}$?

Following Poole [Poo03], an MLN rule is called a parametric factor, or parfactor for short

$\cdot$ Arise in other models, e.g., Probabilistic Soft Logic [BMG10]
Example Revisited

<table>
<thead>
<tr>
<th></th>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>p1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Bob</td>
<td>Anna</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Chloe</td>
<td>Bob</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

4: \((\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y)\)

2: \((\text{OperaFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{OperaFan}(y)\)

8: \(\text{Soccer}(x) \leftrightarrow \neg(\text{OperaFan}(x))\)
What are the factors here?

<table>
<thead>
<tr>
<th>Follows</th>
<th>SoccerFan</th>
<th>OperaFan</th>
</tr>
</thead>
<tbody>
<tr>
<td>p1</td>
<td>p</td>
<td>p</td>
</tr>
<tr>
<td>Anna</td>
<td>Bob</td>
<td>Anna</td>
</tr>
<tr>
<td>Bob</td>
<td>Chloe</td>
<td>Bob</td>
</tr>
<tr>
<td>Anna</td>
<td>Chloe</td>
<td>Chloe</td>
</tr>
</tbody>
</table>

4: \( \text{SoccerFan}(x) \land \text{Follows}(y, x) \rightarrow \text{SoccerFan}(y) \)
2: \( \text{OperaFan}(x) \land \text{Follows}(y, x) \rightarrow \text{OperaFan}(y) \)
8: \( \text{Soccer}(x) \leftrightarrow \neg \text{OperaFan}(x) \)
Core MLN Tasks

- MLNs entail several computational tasks
  - Estimating marginal probability
    - \( z = (x, y, v) \), and given \( x \) and \( y \), compute \( \Pr(y | x) \)
  - Maximum A-Posteriori (MAP) inference
    - \( z = (x, y) \), and given \( x \), compute \( \arg\max_y \Pr(y|x) \)
  - Weight learning: given examples of domains and worlds, find the most likely rule weights to produce the worlds (fitting)
  - Structure learning: given examples of domains and worlds, find “good” rules and weights that are likely to produce the worlds

- These tasks are usually intractable, and common techniques include heuristic local search (MaxWalkSat), sampling (MCMC), gradient-based optimization, message passing, and symmetry-based simplification (lifted inference)
Jha and Suciu [JS12] show how to translate MLNs to tuple-independent databases

I will show a simplified version of that translation on our example
Translation Example (1)

Rule $r(x, y)$:

$$4: \ (\text{SoccerFan}(x) \land \text{Follows}(y, x)) \rightarrow \text{SoccerFan}(y)$$

1. Initialize an empty TID $I$
2. Insert into $I$ each ground fact with probability $1/2$
   - SoccerFan(Anna):0.5, Follows(Bob,Anna):0.5,...
3. For each grounding $r(a, b)$ of $r(x, y)$ add a new fact $R(a, b)$ with probability $4/5$

Define:

$$Q_r() \ := \ \exists x, y \left[ (R(x, y) \land \neg r(x, y)) \lor (\neg R(x, y) \land r(x, y)) \right]$$
Translation Example (2)

- Claim: Our MLN defines the probability distribution \([I]\), restricted to the MLN facts and conditioned on \(\neg Q_r()\).
- Proof idea: view a possible world of \(I\) as composing of two components: MLN facts \((W)\) and \(R\)-facts \((W_R)\).
  - Note: every \(W\) has the same probability in \(I\).
- By using Bayes rule we get:

\[
Pr(W \mid \neg Q_r) = \frac{Pr(\neg Q_r \mid W) \times Pr(W)}{Pr(\neg Q_r)} \sim Pr(\neg Q_r \mid W)
\]
Translation Example (3)

- Let us look at $\Pr(\neg Q_r \mid W)$
- This is the probability that every $R(a, b)$ exists if and only if $r(a, b)$ holds
- Hence:

$$
\Pr(\neg Q_r \mid W) = \Pr(\bigwedge_{W \models r(a, b)} R(a, b) \land \bigwedge_{W \models \neg r(a, b)} \neg R(a, b) \mid W)
$$

$$
= \left( \prod_{W \models r(a, b)} \frac{4}{5} \right) \times \left( \prod_{W \models \neg r(a, b)} \frac{1}{5} \right) \sim \prod_{W \models r(a, b)} 4 \prod_{W \models \neg r(a, b)} 1
$$

$$
= \prod_{W \models r(a, b)} 4
$$
Translating Query Evaluation

- The probability of a query $Q$ over the MLN is then:

$$
Pr(I \models Q \mid I \models \neg Q_r) = \frac{Pr(I \models Q \land \neg Q_r)}{Pr(I \models \neg Q_r)}
$$

$$
= \frac{Pr(I \models Q) - Pr(I \models Q \land Q_r)}{1 - Pr(I \models Q_r)}
$$
Table of Contents

1 Introduction

2 Markov Logic Networks

3 Probabilistic XML
XML Example

```xml
<aerial-photo>
  <region>
    <neighborhood>
      <house size="s">
        <vehicle type="private">
          <neighborhood>
            <factory>
              <facility/>
              <heliport/>
            </factory>
          </neighborhood>
        </vehicle>
      </house>
      <vehicle type="private">
        <neighborhood>
          <factory>
            <facility/>
            <heliport/>
          </factory>
        </neighborhood>
      </vehicle>
    </neighborhood>
  </region>
</aerial-photo>
```
Probabilistic XML (p-Document) Example

XML
Probabilistic Representation
Some Studied Problems

---

Introduction
Markov Logic Networks
Probabilistic XML
References

---

region
aerial photo

neighborhood
house
house
vehicle

factory
facility
parking lot
heliport

size
size
type
track
private

m
s
The p-Document Model

- A *d-document* is a tree with two types of nodes
  - An *ordinary* node is an XML node with a label (or text)
  - A *distributional* node is an encoding of a probabilistic choice of a random set of children
- The *root* of a p-document is always an ordinary node
- There are various types and representations of distributional nodes:
  - Independent choice of children, each child has an associated probability
  - Mutually-exclusive choice, children probabilities sum up to (at most) one
  - Shared random variables (correlations, similar to pc-tables)
Sampling Example

```
<region>
  <aerial photo/>
  <neighborhood>
    <house size="s"/>
    <house size="m"/>
    <vehicle type="private"/>
    <track/>
  </neighborhood>
  <factory>
    <facilities>
      <parking lot/>
      <heliport size="s" private/>
    </facilities>
  </factory>
</region>
```

```
<region>
  <aerial photo/>
  <neighborhood>
    <factory>
      <facilities>
        <heliport size="s" private/>
      </facilities>
    </factory>
  </neighborhood>
</region>
```
A p-document represents a probability space over ordinary XML documents.

To defined this probability space, we need to explain how a random document is being generated/sampled.

Top-down process; for each distributional node $v$:

1. Randomly select a subset of $v$’s children (according to the local distribution of $v$).
2. Throw away unchosen children (and their subtrees).
3. Eliminate $v$ by connecting the chosen children directly to $v$’s parent.
Some Studied Problems

- **Data integration** with probabilistic XML [vKdKA05]
- Evaluating *XPath queries* over p-documents [KKS09, ACK+11]
- Computing the probability of *DTD compatibility* [CKS09]
  - More generally, the probability of acceptance by a *tree automaton*
- Probabilistic XML defined by *probabilistic context-free grammars* (PCFGs) [BKOS10, CK10]
- *Keyword search* over probabilistic XML [LLZW11]


End of lecture 12

More on Probabilistic Databases