Relations, generalizations and the reference-class problem: A logic programming / Bayesian perspective

David Poole

Joint work with Michael Chiang.

University of British Columbia
Overview

➤ Learning from relations: the reference class problem
➤ Inductive logic programming
➤ Probabilities and logic programs: aggregating vs quantifying
➤ Hierarchical priors
➤ Putting it all together
Relational Learning: Example

Given a database containing the relations:

- $\text{grade}$(Student, Course, Grade)
- $\text{dept}$(Course, Department)
- $\text{level}$(Course, Year)
- $\text{major}$(Student, Department)
- $\text{year}$(Student, Year)
- ...

Predict the value (distribution) of $G$ for: $\text{grade}$(joe, cs322, $G$)
Where do the probabilities come from?

➢ To get probabilities from data, you need to aggregate.

➢ To get distribution for grade(joe, cs322, G)
  ➢ distribution of grades of Joe over all courses
  ➢ distribution of grades for all students in CS322
  ➢ distribution of grades for all students over all courses

➢ reference class problem
  ➢ as you generalize you get better statistics, but less specificity
  ➢ conventional wisdom: choose narrowest reference class with adequate statistics
Inductive Logic Programming

{pass(joe,C)}

{pass(S,C) ← dept(C,D) & major(S,D)}

{pass(S,C) ← dept(C,D) & level(C,Y) & major(S,D) & year(S,Y+2)}

{pass(S,C) ← dept(C,cs) & level(C,3) & major(S,cs) & year(S,5)}

{pass(joe,C) ← dept(C,cs)}

{pass(S,cs322) ← major(S,cs) & year(S,5)}

{pass(joe,cs322)}

{pass(S,C) ← dept(C,cs) & C ≠ cs311}

{pass(joe,C) ← dept(C,cs) & level(C,3)}

{}
Adding probabilities to logic programs

Simplest way:

➤ add exogenous “choices of nature” that have probabilities

➤ logic programs give consequences of choices

➢ logic programs have standard syntax and semantics

➢ it suffices to have independent choices

➢ these can represent any belief network:
  local transformation that doesn’t increase the number of parameters
Independent Choice Logic

- **C**, the choice space is a set of alternatives.
  - An **alternative** is a set of atomic choices.
  - An **atomic choice** is a ground atomic formula.
  - An atomic choice can only appear in one alternative.

- **F**, the facts is an acyclic logic program.
  - No atomic choice unifies with the head of a rule.

- **$P_0$** a probability distribution over alternatives:
  $$\forall A \in C \sum_{a \in A} P_0(a) = 1.$$
Meaningless Example

\[ C = \{ \{c_1, c_2, c_3\}, \{b_1, b_2\} \} \]

\[ F = \{ f \leftarrow c_1 \land b_1, \quad f \leftarrow c_3 \land b_2, \]
\[ d \leftarrow c_1, \quad d \leftarrow \overline{c}_2 \land b_1, \]
\[ e \leftarrow f, \quad e \leftarrow \overline{d} \}\]

\[ P_0(c_1) = 0.5 \quad P_0(c_2) = 0.3 \quad P_0(c_3) = 0.2 \]

\[ P_0(b_1) = 0.9 \quad P_0(b_2) = 0.1 \]
Semantics of ICL

➤ A **total choice** is a set containing exactly one element of each alternative in $C$.

➤ For each total choice $\tau$ there is a **possible world** $w_\tau$.

➤ Proposition $f$ is **true** in $w_\tau$ (written $w_\tau \models f$) if $f$ is true in the (unique) stable model of $F \cup \tau$.

➤ The probability of a possible world $w_\tau$ is

$$
\prod_{a \in \tau} P_0(a).
$$

➤ The **probability** of a proposition $f$ is the sum of the probabilities of the worlds in which $f$ is true.
Meaningless Example: Semantics

There are 6 possible worlds:

\[\begin{align*}
  w_1 & \models c_1 b_1 f d e & P(w_1) = 0.45 \\
  w_2 & \models c_2 b_1 \overline{f} d e & P(w_2) = 0.27 \\
  w_3 & \models c_3 b_1 \overline{f} d \overline{e} & P(w_3) = 0.18 \\
  w_4 & \models c_1 b_2 \overline{f} d e & P(w_4) = 0.05 \\
  w_5 & \models c_2 b_2 f \overline{d} e & P(w_5) = 0.03 \\
  w_6 & \models c_3 b_2 f \overline{d} e & P(w_6) = 0.02
\end{align*}\]

\[P(e) = 0.45 + 0.27 + 0.03 + 0.02 = 0.77\]
Logical variables ≡ plates

➤ In logic programming, logical variables are universally quantified

➤ A program means its grounding; multiple instances, one for each individual

➤ Buntine’s plates: parametrized parts of belief networks
Example: Multi-digit addition

```
x_j \cdot \cdots \cdot x_2 \cdot x_1
+ y_j \cdot \cdots \cdot y_2 \cdot y_1
\hline
z_j \cdot \cdots \cdot z_2 \cdot z_1
```

Student Time
- knows carry
- knows addition

Digit Problem
- carry
- \(x\)
- \(y\)
- \(z\)
Rules for multi-digit addition

\[ z(D, P, S, T) = V \leftarrow \]

\[ x(D, P) = Vx \wedge \]

\[ y(D, P) = Vy \wedge \]

\[ carry(D, P, S, T) = Vc \wedge \]

\[ \text{knowsAddition}(S, T) \wedge \]

\[ \text{noMistake}(D, P, S, T) \wedge \]

\[ V \text{ is } (Vx + Vy + Vc) \text{ div } 10. \]

\[ \forall DPST\{\text{noMistake}(D, P, S, T), \text{mistake}(D, P, S, T)\} \in C \]

\[ \forall DPST\{\text{selectDig}(D, P, S, T) = V \mid V \in \{0..9\}\} \in C \]
Plates for Learning

Example: parameter estimation for probability of heads (from [Buntine, JAIR, 94])

\[ \text{heads}_1 \rightarrow q \]
\[ \text{heads}_2 \rightarrow q \]
\[ \ldots \]
\[ \text{heads}_N \rightarrow q \]
ICL Version of Parameter Learning

\[ \text{heads}(C) \leftarrow \text{turns\_heads}(C, \Theta) \land \text{prob\_heads}(\Theta). \]

\[ \text{tails}(C) \leftarrow \text{turns\_tails}(C, \Theta) \land \text{prob\_heads}(\Theta). \]

\[ \forall C \forall \Theta \{ \text{turns\_heads}(C, \Theta), \text{turns\_tails}(C, \Theta) \} \in C \]

\[ \{ \text{prob\_heads}(\theta) : \theta \in [0, 1] \} \in C \]

\[ \text{Prob}(\text{turns\_heads}(C, \theta)) = \theta \]

\[ \text{Prob}(\text{turns\_tails}(C, \theta)) = 1 - \theta \]

\[ \text{Prob}(\text{prob\_heads}(\theta)) = 1 \quad \text{uniform on } [0, 1]. \]
Explaining Data

If you observe:

\[ \text{heads}(c_1), \text{tails}(c_2), \text{tails}(c_3), \text{heads}(c_4), \text{heads}(c_5), \ldots \]

For each \( \theta \in [0, 1] \) there is an explanation:

\[
\{ \text{prob\_heads}(\theta), \text{turns\_heads}(c_1, \theta), \text{turns\_tails}(c_2, \theta), \text{turns\_tails}(c_3, \theta), \text{turns\_heads}(c_4, \theta), \text{turns\_heads}(c_5, \theta), \ldots \}
\]
Aggregating versus quantifying

Consider the difference between:

➤ The distribution of grades for all students in all courses
➤ For all students, the distribution of grades in all courses
➤ For all courses, the distribution of grades over all students
➤ For all students and all courses, the distribution of grades for that student in that course
Quantifying and Aggregating in ICL

➤ for all students, use distribution of grades over all courses
\[ F = \{ \text{grade}(S, C, G) \leftarrow grSt(S, G) \} \]
\[ C = \{ \{ \text{grSt}(S, G) \mid G \in [0, 100] \} \mid S \text{ is a student} \} \]

➤ for all courses, use distribution of grades over all students
\[ F = \{ \text{grade}(S, C, G) \leftarrow grC(C, G) \} \]
\[ C = \{ \{ \text{grSt}(C, G) \mid G \in [0, 100] \} \mid C \text{ is a course} \} \]
Probabilistic Inductive Logic Programming

➤ Given a dataset, choose the best probabilistic logic program given the data... taking into account:

➢ fit to the data

➢ prior probability of the program
Probabilistic Inductive Logic Programming

Given a dataset, choose the best probabilistic logic program given the data... taking into account:

- fit to the data
- prior probability of the program

Is there an alternative?

Bayesian: don't choose the best model, but have a probability distribution over the models

combine all of the models
Probabilistic Inductive Logic Programming

Given a dataset, choose the best probabilistic logic program given the data... taking into account:

➢ fit to the data
➢ prior probability of the program

Is there an alternative?

➢ Bayesian: don’t choose the best model, but have a probability distribution over the models
➢ combine all of the models
Issues with Bayesian ILP

➤ Need to use all of the reference classes; even the most general one!

➤ The lowest reference classes will all have very few observed instances.

➤ Need to use more general reference classes to get the prior on the more specific.

➤ Need a way to combine different most specific reference classes.
Specificity and Counts

\{pass(joe, C)\} \rightarrow \{pass(S, C) \leftarrow dept(C, D) \& major(S, D)\} \rightarrow \{pass(S, C) \leftarrow dept(C, D) \& level(C, Y) \& major(S, D) \& year(S, Y+2)\} \rightarrow \{pass(S, C) \leftarrow dept(C, cs) \& level(C, 3) \& major(S, cs) \& year(S, 5)\} \rightarrow \{pass(S, cs322) \leftarrow major(S, cs) \& year(S, 5)\} \rightarrow \{pass(joe, cs322)\} \rightarrow \{\}

100000
30000
80
25
10
20
3
19
30
230
320
100000
30000
80
25
10
20
3
19
30
230
320
© David Poole 2001
Using The Most General Reference Class
Inferring distributions from generalizations

Even if you knew the distribution of immediate generalizations, how do you infer the appropriate distribution?

\{\text{pass}(S,\text{cs322}) \leftarrow \text{major}(S,\text{cs}) \& \text{year}(S,5)\}\n
\{\text{pass}(\text{joe},C) \leftarrow \text{dept}(C,\text{cs}) \& \text{level}(C,3)\}\n
\{\text{pass}(\text{joe},\text{cs322})\}\
Other Sorts of Rules

\[\text{passed}(S, C) \leftarrow \text{passed}(S, C') \land \text{similarCourses}(C, C').\]

\[\text{passed}(S, C) \leftarrow \text{passed}(S', C) \land \text{similarStudents}(S, S').\]

\[\Rightarrow \text{Collaborative Filtering}\]

Can we also use the same technique to learn similar grades?
Lessons from history

➤ In the Seventeenth century, there were accurate models predicting the motion of stars and planets using universal function approximators (epicycles).

➤ Even when Newton came up with the “correct” model, it took a long time to fit the data as well.

➤ We need representations that can express the “correct” models, even if these may be difficult to find.
Conclusion

➤ Mix of logic programming + Bayesian learning seems to be most promising

➤ Many problems still to be solved

➢ some, such as the *reference class problem*, have a long history

➢ some are new

➢ the combination is relatively unexplored

➤ You can anticipate many different solutions