KU LEUVEN

Scalable Rule Learning in Probabilistic Knowledge Bases

SafeLearner

Knowledge Bases (KBs) are becoming increasingly:

- Larger
- Probabilistic
- Incomplete

What is new?
To do completion of uncertain knowledge bases, we use Lifted Probabilistic Inference

Why *SafeLearner*?

- Significantly faster than ProbFOIL+
- Scales as good as AMIE+

How much/fast does it scale?
SafeLearner has demonstrated the learning of rules on a KB with 14k+ tuples under 2.5 hours

Problem specification

Given:

- A probabilistic KB (PDB) : D
- A target predicate : $target$ of arity k
- Loss Function : L
- Set of $target$ examples : $E = \{\langle t_i, p_i\rangle | i = 1, \dots M\}$

To Find: $H^* \in L_H$ such that it minimizes cross-entropy

$$H^* = \underset{H}{\operatorname{argmax}}\, Cross\ Entropy\ (H, E, D)$$
$$= \underset{H}{\operatorname{argmin}} \sum_{\langle t_i, p_i\rangle \in E} (p_i \log q_i + (1 - p_i) \log(1 - q_i))$$

where q_i is the predicted probability of ith atom t_i.

What are Probabilistic Databases?

In any database, if one considers all the tables with n columns as *predicates* with arity *n*, then one can construct a set of *atoms* where each *atom* is a tuple of the form *predicate(constants)*. Assigning a probability value to each atom in a database will form a PDB.

Thus, a PDB is a set of tuples of form $\langle atom, probability\rangle$.

Example

researcher	researcher	p
alice	edwin	0.2
alice	fred	0.3
bob	carl	0.4
bob	greg	0.5
bob	harry	0.6
bob	ian	0.7
carl	greg	0.8
carl	harry	0.9
carl	ian	0.8
dave	edwin	0.7
dave	fred	0.6
edwin	fred	0.5
greg	harry	0.4
greg	ian	0.3
ian	ian	0.2

$target$: coauthor/2

researcher	paper	p
bob	plp	0.9
carl	plp	0.6
greg	plp	0.7
ian	db	0.9
harry	db	0.8

author/2

researcher	university	p
edwin	harvard	1.0
fred	harvard	0.9
alice	mit	0.6
dave	mit	0.7

location/2

Learned Rules (H^*):

- 0.241::coauthor(A, B) :- location(A, C), location(B, C).
- 0.992::coauthor(A, B) :- author(A, C), author(B, C).

Key features

- Uses **Lifted Inference** in rule learning, thereby avoiding the use of grounding for knowledge compilation
- Uses **memoization**/caching to store the canonical structures of all the queries with their probability expressions
- Breaks larger queries into independent subqueries for better performance
- Uses Sequential Least SQuares Programming (SLSQP) Algorithm to simultaneously optimize probabilistic weights of all the rules in the hypothesis
- Uses the input format of $ProbFOIL^+$

Algorithm 1 SafeLearner – Main Algorithm

Input: PDB $\mathcal{D}$, $target$, loss $\mathcal{L}$
E := Set of all $target$ tuples in $\mathcal{D}$
H := Set of all the type consistent and significant (deterministic) rules from AMIE+ using $\mathcal{D}$ with $target$ in head
Initialize probability p_{h_i} for each rule h_i in H
Embed rule probabilities $p_h = \{p_{h_1}, p_{h_2}, \dots, p_{h_n}\}$ in H
Sample $target$ tuples in E_2 and E_3 and compute their respective sampling weights w_2 and w_3
Q := QueryConstructor(H) ▷ Q is safe
for i in range(0, MaxIterations) **do**
  Randomly select an example e from all the target examples $E \cup E_2 \cup E_3$
  y := ProbabilityPredictor(Q, e) ▷ y is a function of p_h
  if $e \in E$ **then**
    x := actual probability of e from E
    Compute $\mathcal{L}$ for e using x, y with sampling weight := 1
  else if $e \in E_2$ **then**
    Compute $\mathcal{L}$ for e using $x := 0$ and y with sampling weight := w_2
  else if $e \in E_3$ **then**
    Compute $\mathcal{L}$ for e using $x := 0$ and y with sampling weight := w_3
  Get gradient of $\mathcal{L}$ at current p_h
  Update p_h
Remove rules with insignificant rule weights from H
return H

Reference: https://openreview.net/pdf?id=HkyI-5667, 1st conference on Automated KB Construction 2019

Arcchit Jain – KU Leuven
Tal Friedman – University of California Los Angeles
Ondřej Kuželka – KU Leuven
Guy Van den Broeck – University of California Los Angeles
Luc De Raedt – KU Leuven

erc
European Research Council
Established by the European Commission