Efficient relational learning from sparse data

Luboš Popelínský
Knowledge Discovery Group
Faculty of Informatics, Masaryk University in Brno, Czechia
popel@fi.muni.cz
http://www.fi.muni.cz/kd

Relational learning - learning in first-order logic
Exact learning - learning from exact data
Sparse data - not more than 5 training examples
Generate & test top-down algorithms - from the most general hypothesis
Assumption-based learning

BK, E, bias, A=true

inductive engine

fails program P
assumption A

A acceptable?
true return(P)

fails

generate assumption

assumption A
Generic algorithm of assumption-based learning

Given:

domain knowledge $BK$, example set $E$, bias,
assumption $A = true$
inductive engine $I$, overgeneral program $P$
function $f$, that computes an assumption $A$
acceptability module $AM$

1. Call $I$ on $BK \cup P, E \cup A, bias$.
   - **if** $I$ succeeds resulting in program $P'$
     - **then** call $AM$ to validate the assumption $A$.
       - **if** $A$ is accepted **then** return($P'$) **else** go to (2).
     - **else** go to (2).

2. Call $f$ to generate a new assumption $A$. If it fails, return(fail) and stop else go to (1).
WiM

inductive engine *Markus*\(^+\)

- depth-first search
- automatic setting of bias
- multiple predicate learning
- 2nd-order schema may be employed

generator of assumptions

- choose the simplest positive example
- find its *near-miss*

acceptability criterion

- membership oracle
WiM: results

2 – 4 examples for learning most of ILP benchmark predicates (list processing, Peanova aritmetika)

learning from positive examples only; negative examples, if any, generated with WiM itself

max. 1 query to the user

less dependent on quality of examples

easy to use
**CRUSTACEAN**, **SKILit** a **WiM** : Randomly generated examples

<table>
<thead>
<tr>
<th></th>
<th><strong>CRUSTACEAN</strong></th>
<th><strong>SKILit</strong></th>
<th><strong>WiM</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>member</td>
<td>0.65 0.76</td>
<td>0.70 0.89 0.95</td>
<td>0.80 0.97 0.97</td>
</tr>
<tr>
<td>last</td>
<td>0.74 0.89</td>
<td>0.71 0.72 0.94</td>
<td>0.76 0.89 0.94</td>
</tr>
<tr>
<td>append</td>
<td>0.63 0.74</td>
<td>0.76 0.80 0.89</td>
<td>0.77 0.95 0.95</td>
</tr>
<tr>
<td>delete</td>
<td>0.62 0.71</td>
<td>0.75 0.88 1.00</td>
<td>0.85 0.88 0.97</td>
</tr>
<tr>
<td>reverse</td>
<td>0.80 0.86</td>
<td>0.66 0.85 0.87</td>
<td>0.85 0.95 0.99</td>
</tr>
</tbody>
</table>
Randomly generated examples: Learning with assumptions

<table>
<thead>
<tr>
<th># pos.</th>
<th>2</th>
<th>3</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>bez</td>
<td>s</td>
<td>TP</td>
</tr>
<tr>
<td>last</td>
<td>0.885</td>
<td>0.896</td>
<td>6</td>
</tr>
<tr>
<td>delete</td>
<td>0.882</td>
<td>0.962</td>
<td>8</td>
</tr>
<tr>
<td>leq</td>
<td>0.380</td>
<td>0.703</td>
<td>0</td>
</tr>
<tr>
<td>length</td>
<td>0.540</td>
<td>0.659</td>
<td>0</td>
</tr>
</tbody>
</table>
DWiM schema

database schema and object descriptions in F-logic

\[ \text{GENERATE} \]

Learning set \quad \text{Domain knowledge predicates}

\[ \text{WiM} \]

the new class/attribute definition in FOL

\[ \text{TRANSLATE} \]

the new class/attribute definition in F-logic
Spatial database schema

BRIDGE
- Object 1
- Object 2

LINEAR
- Geometry

RAILWAY
- Named
  - State
  - Importance

PLANAR
- Geometry
  - FORESTRY
    - FOREST
    - WOOD
  - BUILDING
    - FOREST HOUSE
  - Forest
Inductive query language for mining in geographic data
[PKDD’98]

extract characteristic rule
for bridge
from road, river.

extract discriminate rule
for forest
in contrast to wood
from point of view area.

bridge(X,Y):-
    road(X),roadGeometry(X,Z),
    river(Y),riverGeometry(Y,U),
    member(V,Z),member(W,U),W=V.

forest(F) :-
    geometry(F,GForest),
    area(GForest,Area),
    100 < Area.
extract dependency rule

for differentHouses

from forestHouse, forest, building

where building(B, GB),
    not forestHouse(B, F)

from point of view distance, less