Meta-interpretive learning of data transformation programs

Andrew Cropper, Alireza Tamaddoni-Nezhad, Stephen H. Muggleton
Imperial College London
<table>
<thead>
<tr>
<th>ID</th>
<th>Age</th>
<th>Diagnosis</th>
<th>Lung Disease</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_011</td>
<td>67</td>
<td>Unknown</td>
<td>n/a</td>
<td>80.78%</td>
</tr>
<tr>
<td>P_003</td>
<td>56</td>
<td>Carcinoma, lung disease: unknown</td>
<td>carcinoma, unknown</td>
<td>20.78%</td>
</tr>
<tr>
<td>P_013</td>
<td>70</td>
<td>Pneumonia</td>
<td></td>
<td>55.9</td>
</tr>
</tbody>
</table>

- Semi-structured
- Positive only learning
- Background knowledge
<table>
<thead>
<tr>
<th>ID</th>
<th>Age</th>
<th>Diagnosis</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_011</td>
<td>67</td>
<td>Lung disease: n/a, Diagnosis: <strong>Unknown</strong></td>
<td>80.78%</td>
</tr>
<tr>
<td>P_003</td>
<td>56</td>
<td><strong>Carcinoma</strong>, Lung disease: unknown</td>
<td>20.78%</td>
</tr>
<tr>
<td>P_013</td>
<td>70</td>
<td><strong>Pneumonia</strong></td>
<td>55.9%</td>
</tr>
</tbody>
</table>

Input

```
f(A,B):- f2(A,C), f1(C,B).
f2(A,B):- find_patient_id(A,C), find_int(C,B).
f1(A,B):- open_interval(A,B,[':',' '],['','n']).
f1(A,B):- open_interval(A,B,[':',' '],['',' ']).
```
MetagolD

Implementation of meta-interpretive learning*, a form of inductive logic programming based on a Prolog meta-interpreter, which supports predicate invention and the learning of recursive theories

Transformation language

- find_sublist/3
- closed_interval/4
- open_interval/4
open_interval/4 and closed_interval/4

Input = [i,n,d,u,c,t,i,o,n],
Start = [n,d],
End = [t,i]

open_interval(Input,[u,c],Start,End).
closed_interval(Input,[n,d,u,c,t,i],Start,End).
Harpalus rufipes eats large prey such as Lepidoptera Bembidion lampros. In cereals the main food was Collembola.

Learned program:

```
f(A,B):- f3(A,C), find_species(C,B).
f3(A,B):- find_species(A,C), f2(C,B).
f2(A,B):- closed_interval(A,B,[f,o],[o,d]).
f3(A,B):- find_species(A,C), f1(C,B).
f1(A,B):- closed_interval(A,B,[e,a],[t,s]).
```
Experiment: ecological papers

No. training examples

Mean predictive accuracy

No. training examples

Mean learning time (seconds)

- delimiter size 1
- delimiter size 2
- delimiter size 3
- default accuracy
Experiment: medical records

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>P_011 67 year lung disease: n/a, Diagnosis: <strong>Unknown</strong> 80.78%</td>
<td>P_011 67 <strong>Unknown</strong></td>
</tr>
<tr>
<td>P_003 56 Diagnosis: <strong>carcinoma</strong>, lung disease: unknown 20.78</td>
<td>P_003 56 <strong>carcinoma</strong></td>
</tr>
<tr>
<td>P_013 70 Diagnosis: <strong>pneumonia</strong> 55.9</td>
<td>P_013 56 <strong>pneumonia</strong></td>
</tr>
</tbody>
</table>

f(A,B):- f2(A,C), f1(C,B).
f2(A,B):- find_patient_id(A,C), find_int(C,B).
f1 (A,B):- open_interval(A,B,[':',' '],['','n']).
f1 (A,B):- open_interval(A,B,[':',' '],['',''])
Experiment: medical records

![Graph showing the relationship between training examples and mean predictive accuracy and learning time](image)

- **Mean predictive accuracy**
  - No. training examples: 1, 2, 3, 4, 5
  - Delimiter sizes: 1, 2, 3
  - Default accuracy

- **Mean learning time (seconds)**
  - No. training examples: 1, 2, 3, 4, 5
  - Delimiter sizes: 1, 2, 3
  - Default accuracy

*Experiment: medical records*
Conclusions
• MIL is able to generate accurate data transformation programs from a small number of examples
• Delimiter size effects learning performance

Future work
• Apply to problems which require recursion
• Generate hypotheses in a scripting language
• Probabilistic approaches / noise handling
Thank you