Boosting the Efficiency of First-order Abductive Reasoning Using Pre-estimated Relatedness between Predicates

Kazeto Yamamoto, Naoya Inoue, Kentaro Inui, Yuki Arase and Jun'ichi Tsujii

Tohoku University, Japan
Abduction

Inference to find the best explanation to observations.

- My garden is wet.
- The neighbor’s garden is wet.
- It rained last night.
- The neighbor used a sprinkler.
Abduction

😊 A promising framework for Natural Language Understanding

- Interpretation as Abduction [Hobbs+, ’93]
- Plan Recognition [Ng & Mooney, ‘92]
- Recognizing Textual Entailment [Ovchinnikova, ‘11]
- Coreference Resolution [Inoue & Inui, ’12]
Advantages

😊 Produces interpretable explanations

😊 Abstracts away from process of computation (declarative nature of logic)

😊 and more...

   – Provides a principled way of using knowledge
Abduction

- Given $B$, $O$:
  - Knowledge $B$ : Set of first-order axioms
  - Observation $O$ : conjunctive set of literals

- Output hypothesis (**explanation**) $H^*$ :

Set of literals with a maximum score

$$H^* = \arg \max_{H \in \mathcal{H}} \text{score}(H)$$

where

$$B \cup H \models O$$
$$B \cup H \not\models \bot$$
Abduction

- Given $B, O$:
  - Knowledge $B$ : Set of first-order axioms
  - Observation $O$ : conjunctive set of literals

- Output hypothesis (explanation) $H^*$:
  Set of literals with a maximum score

$$H^* = \arg\max_{H \in \mathcal{H}} \text{score}(H)$$

where

$$B \cup H \models O$$
$$B \cup H \not\models \bot$$

$B$: 300,000+ axioms

$O$: 20 literals

Combinatorial optimization over 1,000 variables

Mini-TACITUS (Mulkar-Mehta+ 07): $\geq 30 \text{ min}$

Emulation on MLNs (Blythe+ 11): $7 \text{ min}$
ILP-based Abduction
[Inoue&Inui, ’11, ’12]

 Translate abductive lifted inference to ILP (Integer Linear Programming)

– Many matured, efficient solvers are applicable (e.g., Gurobi Optimizer, IBM CPLEX)

\[ \begin{align*} B: & \quad 300,000+ \text{ axioms} \\
O: & \quad 20 \text{ literals} \\
\end{align*} \]

\[ \begin{align*}
\text{Mini-TACITUS (Mulkar-Mehta+ 07)}: & \quad \geq 30 \text{ min} \\
\text{Emulation on MLNs (Blythe+ 11)}: & \quad 7 \text{ min} \\
\text{ILP-based & CPI (Inoue+, 12)}: & \quad 0.4 \text{ sec} \\
\end{align*} \]
ILP-based Abduction
[Inoue & Inui, ’11, ’12]

Translate abductive lifted inference to ILP (Integer Linear Programming)

– Many matured, efficient solvers are applicable (e.g., Gurobi Optimizer, IBM CPLEX)

But not enough efficient...

B: 300,000+ axioms
O: 20 literals

Mini-TACITUS (Mulkar-Mehta+ 07): ≥ 30 min
Emulation on MLNs (Blythe+ 11): 7 min
ILP-based & CPI (Inoue+, 12): 0.4 sec

B: 1,000,000+ axioms
O: 30 literals

ILP-based & CPI (Inoue+, 12): ≥ 20 min
The computational cost strongly depends on the size of potential elemental hypotheses...

**Observation**

john(x) ^

**Knowledge Base**

go(e) => buy(...

**Solution Hypothesis**

buy(e) => sell(...

**Potential Elemental Hypotheses**

(= set of literals in possible candidates of solution)
The computational cost strongly depends on the size of potential elemental hypotheses...

Potential Elemental Hypotheses

(= set of literals in possible candidates of solution)
A* search-guided P.E.H generation

John went to the store and bought a book.
Control potential elemental hypotheses generation using $A^*$ search technique

Observation:

John went to the store and bought a book.
John went to the store and bought a book.
A* heuristic function

Knowledge Base

Abstract first-order literals to propositional literals

※Distance function; \( \delta(a) = 1 \)

<table>
<thead>
<tr>
<th></th>
<th>shot/3</th>
<th>have_gun/2</th>
<th>kill/3</th>
<th>die/2</th>
<th>is_sick/2</th>
<th>cough/2</th>
<th>hunter/1</th>
</tr>
</thead>
<tbody>
<tr>
<td>shot/3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>inf</td>
<td>inf</td>
<td>1</td>
</tr>
<tr>
<td>have_gun/2</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>inf</td>
<td>inf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>kill/3</td>
<td>0</td>
<td>1</td>
<td>inf</td>
<td></td>
<td>inf</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>die/2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>is_sick/2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cough/2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>hunter/1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
Experiments

28 literals on average
- the same dataset as the one used by [Inoue & Inui, '12]
- created by converting the development dataset of RTE-2
- 777 observations

289,655 axioms
Resource:
- WordNet
- FrameNet

Systems:
1. ILP+CPI [Inoue&Inui, '12]
2. Our system※

Observations → Knowledge base → Inference Engine

Timeout: 120 sec.

Compare each inference time

※ publicly available at https://github.com/kazeto/phillip
## Results (in average)

<table>
<thead>
<tr>
<th></th>
<th>[Inoue &amp; Inui, '12]</th>
<th>Our system</th>
</tr>
</thead>
<tbody>
<tr>
<td># of literals in P.E.H.</td>
<td>1120</td>
<td>349</td>
</tr>
<tr>
<td># of chains in P.E.H.</td>
<td>1027</td>
<td>302</td>
</tr>
<tr>
<td># of unifications P.E.H.</td>
<td>460</td>
<td>166</td>
</tr>
<tr>
<td>Time (generation) [sec]</td>
<td>0.14</td>
<td>0.13</td>
</tr>
<tr>
<td>Time (conversion) [sec]</td>
<td>0.21</td>
<td>0.07</td>
</tr>
<tr>
<td>Time (solving) [sec]</td>
<td>5.93</td>
<td>1.46</td>
</tr>
<tr>
<td>Time (all) [sec]</td>
<td>6.29</td>
<td>1.67</td>
</tr>
<tr>
<td># of timeout</td>
<td>70</td>
<td>33</td>
</tr>
</tbody>
</table>

Our system improves the computational complexity by reducing the size of elemental potential hypotheses.
Results

Problems which cannot be solved by [I&I 12] within 2 min.

# of problems: 777
Timeout: 120 sec.
**Latest results**

$B$: 1,000,000+ axioms  
$O$: 30 literals

ILP-based & CPI (Inoue+, 12): $\geq$ 20 min
A* search-based: $\geq$ 20 min
Our latest version: 1.3 sec
Chris was running after John, because he wanted to talk to him.
Summary

✓ Abduction has been considered to be the promising framework for many applications
✓ The computational complexity is an important issue in abduction
✓ We proposed a method that makes a significant improvement over the current state-of-the-art system

➢ The latest version of our work is publicly available at http://github.com/kazeto/phillip
Backup!!
The process of interpreting sentences in discourse can be viewed as the process of providing the best explanation of why the sentences would be true.
Interpretation as Abduction

(Hobbs+, 1993)

Background Knowledge (Axioms)

\[
\begin{align*}
\text{loan}(y_2) & \Rightarrow (\exists e_2, y_1, y_3) \text{issue}(e_2, y_3, y_2, y_1) \land \text{financial\_inst}(y_3) \\
\text{issue}(e_2, x_2, x_3, x_1) & \land \text{financial\_inst}(x_2) \Rightarrow \text{go}(e_2, x_1, x_2) \\
\Rightarrow \text{bank}(x_2) & \Rightarrow \text{get}(e_2, y_1, y_2)
\end{align*}
\]

Observation

\((\exists e_1, e_2, x_1, x_2, y_1, y_2)\)

\[
\begin{align*}
\text{John}(x_1) & \land \text{go}(e_1, x_1, x_2) \land \text{bank}(x_2) \land \text{he}(y_1) \land \text{get}(e_2, y_1, y_2) \land \text{loan}(y_2)
\end{align*}
\]

Input

\textit{John went to the bank.  He got a loan.}
John went to the bank. He got a loan.

**Input**

**Observation**

\[ \exists e_1, e_2, x_1, x_2, y_1, y_2 \]
\[ John(x_1) \land go(e_1, x_1, x_2) \land bank(x_2) \land he(y_1) \land get(e_2, y_1, y_2) \land loan(y_2) \]

**Explanation**

- John and he are coreferent
- went to the bank is the purpose of got a loan
- bank refers to a financial bank

\[ \exists e_2, y_1, y_3 \]
\[ issue(e_2, y_3, y_2, y_1) \land financial\_inst(y_3) \]

\[ x_1 = y_1 \]
\[ x_2 = y_3 \]

\[ (\exists e_2, y_1, y_3) \]
\[ issue(e_2, x_2, x_3, x_1) \land go(e_2, x_1, x_2) \land bank(x_2) \land get(e_2, y_1, y_2) \land loan(y_2) \]

**Interpretation as Abduction**

(Hobbs+, 1993)
Problems (in ‘90s)

1. KB was not large enough
   → Knowledge acquisition from Web

2. Inference was not fast enough
   → Developed a fast reasoner

3. Scoring functions were not easy to tune
   → Machine learning is applicable
### Result of pre-estimation

<table>
<thead>
<tr>
<th>Max distance</th>
<th>Time [sec.]</th>
<th>File size</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>106</td>
<td>0.8 GB</td>
</tr>
<tr>
<td>6</td>
<td>1514</td>
<td>5.8 GB</td>
</tr>
<tr>
<td>8</td>
<td>7841</td>
<td>28 GB</td>
</tr>
</tbody>
</table>
Cancel impossible inference

police(Tom) XOR criminal(Tom) are mutual exclusive!!

Knowledge Base

police(x) => have(e1)^gun(y)^nsubj(e1,x)^dobj(e1,y)

have(e2) gun(u2) nsubj(e2,Tom) dobj(u2)

have(e1)^gun(y)^nsubj(e1,x)^dobj(e1,y) => shoot(e2)^nsubj(e2,x)^dobj(e2,z)

shoot(e1) nsubj(e1,Tom) dobj(e1,u1)

shoot(e)^nsubj(e,x)^dobj(e,y) => criminal(x)

criminal(Tom)
Parent-child relationships between literals

- X eat Y
  - eat(e1)
  - nsubj(e1,X)
  - dobj(e1,Y)

- Z cook Y
  - cook(e3)
  - nsubj(e3,Z)
  - dobj(e3,Y)

- X buy Y
  - buy(e2,X)
  - nsubj(e2,X)
  - dobj(e2,Y)
Backward chaining with considering relations between literals

Knowledge Base

- kill(e1) ^ nsubj(e1,x) ^ dobj(e1,y)
  => die(e2) ^ nsubj(e2,y)

- kill(u1) nsubj(u1,u2) dobj(u1,x)
- kill(u3) nsubj(u3,u4) dobj(u3,y) e1=u3
- kill(u5) nsubj(u5,u6) dobj(u5,z) e1=u5
- kill(u7) nsubj(u7,u8) dobj(u7,y) e1=u7

- die(e1) nsubj(e1,x) ... nsubj(e2,y) ... nsubj(e3,z) ... nsubj(e4,y) ...
Backward chaining with considering relations between literals

Knowledge Base

\[ \text{kill}(e1) \land \text{nsubj}(e1,x) \land \text{dobj}(e1,y) \Rightarrow \text{die}(e2) \land \text{nsubj}(e2,y) \]

\[ \text{kill}(u1) \land \text{nsubj}(u1,u2) \land \text{dobj}(u1,x) \]

\[ \text{kill}(u11) \land \text{nsubj}(u11,u12) \land \text{dobj}(u11,y) \]

\[ \text{kill}(u9) \land \text{nsubj}(u9,u10) \land \text{dobj}(u9,y) \]

\[ \text{kill}(u5) \land \text{nsubj}(u5,u6) \land \text{dobj}(u5,y) \]

\[ \text{kill}(u3) \land \text{nsubj}(u3,u4) \land \text{dobj}(u3,y) \]

\[ \text{kill}(u7) \land \text{nsubj}(u7,u8) \land \text{dobj}(u7,y) \]

\[ \text{die}(e1) \land \text{nsubj}(e1,x) \land \text{nsubj}(e2,y) \land \text{nsubj}(e3,z) \land \text{nsubj}(e4,y) \land \ldots \]
Restrict improper unifications

know(e4)  nsubj(e4,x4)  nsubj(e6,x6)  eat(e6)

reply(e3)  nsubj(e3,x3)  nsubj(e5,x5)  sleep(e5)

eat(e1)  nsubj(e1,x1)  nsubj(e2,x2)  go(e2,x2)
Restrict improper unifications

know(e4) nsubj(e4,x4) nsubj(e6,x6) eat(e6)

reply(e3) nsubj(e3,x3) nsubj(e5,x5) sleep(e5)

eat(e1) nsubj(e1,x1) nsubj(e2,x2) go(e2,x2)