Feature Extraction with Description Logics and Functional Subsumption

Rodrigo de Salvo Braz
Dan Roth

University of Illinois at Urbana-Champaign
A conflict

? Most machine learning algorithms use feature vectors as inputs.

? Most data is best represented as structured data.

? Feature extraction is the conversion from one to the other (and may be most of the work).
Structured data – II
Feature Extraction

Structured example

Human-written feature types
Feature Extraction

- Typically done in ad hoc fashion:
  - Prevents general analysis;
  - Prevents Feature Extraction/Learning unified analysis (e.g. kernels).
- Using a language is tricky
  - Type of inference.
  - May be intractable if not careful.
A language for declaring which features to generate

Feature type specifications by directed trees

Example segment
Generating feature vectors

Example

Feature types

Example

Feature types
Generating feature vectors

Example

Feature types
Generating feature vectors

Example

Feature types

1

spouse

child

name(john) male

name(mary) female

name(jill) female

name(peter) male age(40)

name(margot) spouse friend

student child child

child
Generating feature vectors

Example

- Male: name(john)
- Female: name(mary), name(jill), name(jenny)
- Tall: name(mary)
- Age: 40
- Spouse: name(peter)
- Friend: name(margot)
- Student: name(jill)
- Child

Feature types

- Child
- Spouse
- Friend
Generating feature vectors

Example

```
male
name(john)
spouse
female
name(jenny)
spouse
female
name(jill)
spouse
female
name(summer)
spouse
female
name(peter)
```

```
name(mary)
spouse
name(margot)
friend
student
child
```

```
male
tall
```

```
feature
```

Feature types

```
child
male
tall
```

Nothing like this in the example!
Generating feature vectors

**Example**

1. male
   - name(john)
   - spouse
     - male
     - name(mary)
     - spouse
     - female
     - name(jill)
     - student
     - child
     - child
     - child
     - child
     - spouse
     - friend
     - name(margot)

2. male
   - name(peter)
   - age(40)

3. female
   - name(mary)
   - tall

**Feature types**

1. male
2. female
3. child
Generating feature vectors

Example

Feature types
Generating feature vectors

Example

Feature types
Generating feature vectors

Example

Feature types

<table>
<thead>
<tr>
<th>Feature types</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Feature Description Logics

(AND
  (SOME spouse ANY)
  (SOME child (AND male tall)))

(SOME spouse
  (SOME friend female))
Subsumption

A description $C$ subsumes ($\supseteq$) a description $D$ if every individual in $D$ must be in $C$, no matter the interpretation.

Subsumption is tractable.

\[ C = (\text{AND} \quad (\text{SOME spouse ANY}) \quad (\text{SOME child male})) \]
\[ D = (\text{AND} \quad (\text{SOME spouse (SOME student ANY)}) \quad (\text{SOME child (AND tall male)}) \quad (\text{SOME child female})) \]
Feature extraction as subsumption

Feature type

Example

(SOME child female)

(AND
  SOME friend
  (AND
    name(carol)
    SOME child (AND name(kelly) female))
  SOME child name(john))

Description of node
Feature extraction as subsumption

(SOME child female)

Feature type

Example

(name(john))

Description of node

(name(kelly) female)

(name(john))

(name(carol) friend)

(child female)

(child)

(name(kelly) female)

(name(john))
Feature extraction as subsumption

(SOME child female)

(AND name(kelly) female)

Description of node
Feature extraction as subsumption

(SOME child female)

(AND
  name(carol)
  SOME child (AND name(kelly) female))

Description of node
A problem in practice

Subsumption would be natural in this case but does not occur
A problem in practice
A problem in practice

name(schwarzenegger) ⊇ name(schwarzneger)

job

.actor  governor

job

Make comparison more flexible

At core of subsumption algorithm is the comparison of attributes:

... if (attr1 == attr2) ...

We simply make that a function call:

... if (f (attr1, attr2) == 1) ...
Is this just a hack?

What about the nice DL semantics?
Is this just a hack?

What about the nice DL semantics?

In fact, equivalent to “shallow OR” (tractable).
Is this just a hack?

What about the nice DL semantics? In fact, equivalent to “shallow OR” (tractable). Replace any \texttt{attr} by (OR \texttt{a}_1 \texttt{a}_2 \ldots \texttt{a}_n) where \(f(\texttt{attr}, \texttt{a}_i) = 1\).

\texttt{AND kill}
\texttt{(SOME object JFK)}

\texttt{AND (OR kill murder assassinate)}
\texttt{(SOME object (OR JFK kennedy “John F. Kennedy” ...))}
Why not just use shallow OR then?

- Function is an implicit representation.
- We may incorporate procedural knowledge:
  - Typos;
  - Similar sounding words;
  - Context-sensitive knowledge.
Take home message

- Feature Description Logics provides an expressive way to deal with structured examples.
- Syntax choices render it tractable.
- Allows for FE-learning integrated approaches like kernels (Cumby & Roth 2003).
- Can be made even more expressive with little extra cost by functional subsumption.
The End