

A Semantic Memory for Incremental Ontology Population

Berenike Loos and Lasse Schwarten

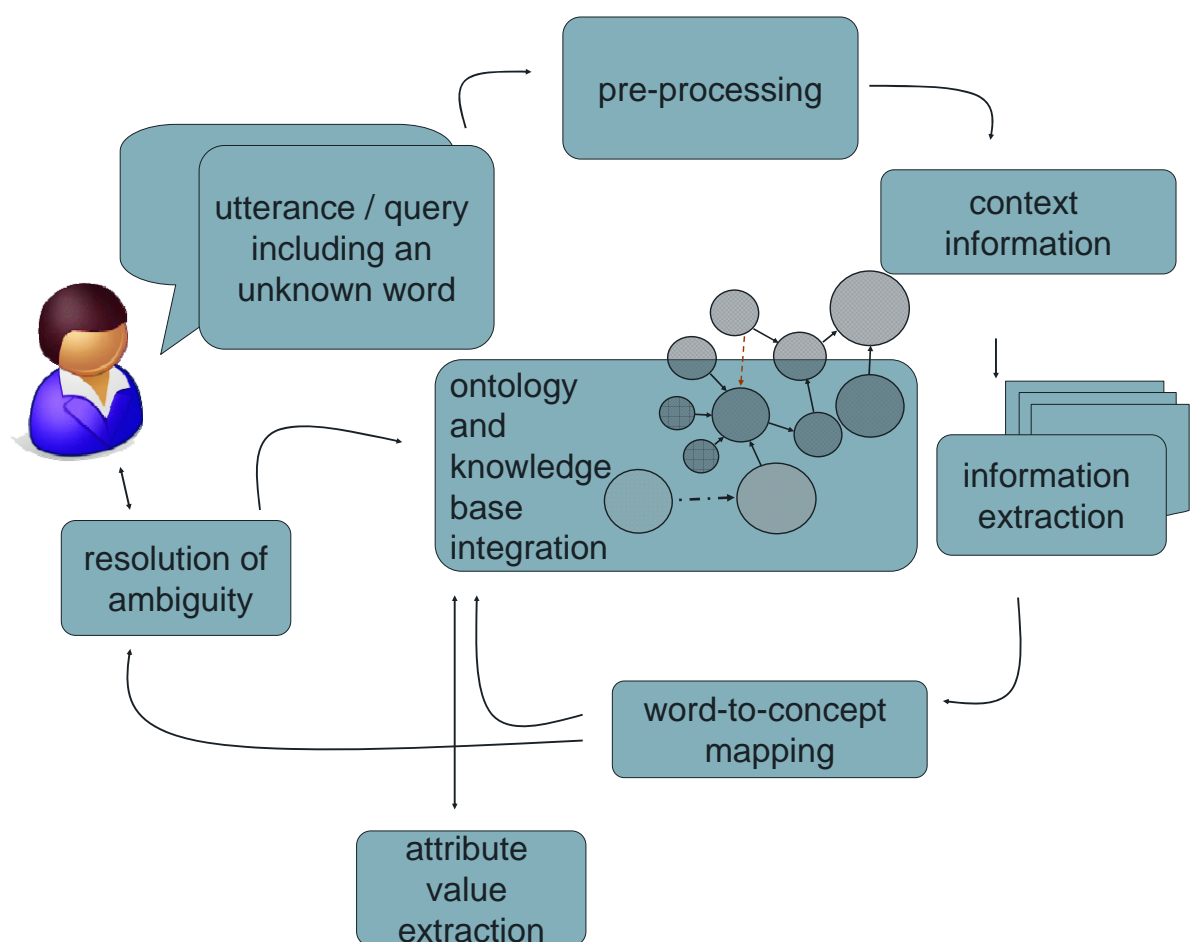
German National Library - Information Technology - Adickesallee 1, D-60322 Frankfurt am Main - b.loos@d-nb.de
University of Bremen - Computer Science - Bibliothekstr. 1, D-28359 Bremen - lasse@tzi.de

Motivation

- Generally, ontology learning and population is applied as a semi-automatic approach to knowledge acquisition in natural language understanding systems. After the ontology is created or populated, an expert of the domain can still change or refine the newly acquired knowledge.
- In an incremental ontology learning and population framework this approach is not sufficient as knowledge about the real world is dynamic and, therefore, has to be acquired and updated constantly.
- We propose the storing of newly acquired instances of an ontological concept in a separate database instead of integrating them directly into the system's knowledge base. The advantage is that incorrect knowledge is not part of the internal knowledge representation but stored aside. Furthermore, information about the confidence of the learned instances can be displayed and used for a final revision by an expert as well as for a further fully automatic acquisition.

Scenario

1. The user of a natural language understanding system tries to get information about a place which is not included in the knowledge base of the system so far: E.g. he asks "How do I get to the i-Punkt?".
2. The unknown term is the instance "i-Punkt", therefore, the appropriate concept in the ontology might be *Bar*.
3. This concept inherits a *has-street-name* property (as well as many others) from a superclass *Building*. With the help of this hint and a contextual specification with respect to the location of the user, the corresponding street name of the place can be retrieved on the Internet with information extraction methods.
4. Now the newly acquired knowledge is given a value to express its confidence about the extracted information and the mapping of the found hypernym into an ontological class, to which the instance can be added by an *instance-of* relation.
5. The automatically acquired knowledge can be reviewed by an expert in an orderly manner as he sees the confidence values of the system and can change the database entries without having complete knowledge about the ontology.



```
SELECT * FROM instance_learning.instance_learning_table i;
```

OOV	Context	SemanticClass	Confidence	Instance_id
Auerstein	Heidelberg	Restaurant	0.9	2
Bolero	Bremen	Bar	1	4
Brommy	Bremen	Bar	0.6	5
i-Punkt	Heidelberg	Bar	0.7	1

```
SELECT * FROM instance_learning.semanticproperties s;
```

name	value	confidence	instance_id
has-country-code	49	0.5	1
has-area-code	6221	0.5	1
has-house-number	30	0.5	1
has-postal-code	69117	0.7	1
has-position-quality	LowPosition	0.5	1
has-street-name	Untere Str.	0.7	1
has-hompage	www.i-punkt-heidelberg.de	1	1
has-city	Heidelberg	0.9	1
has-phone-number	602441	0.7	1
has-happy-hour	Mo: Kölsch 1,00 € oder Havana-Cola 3,8...	0.3	1
has-special-offer	Mo: Deutsche Pop und Rock Musik	0.3	1

The Semantic Memory

- Whenever the system encounters an unknown term during a user's inquiry it tries to request the semantic memory by giving the term and the context location (as e.g. Heidelberg or Bremen) in which it appears. If it is successful, the system can utilize the retrieved information and continue. In this case the semantic memory acts as a cache which increases the overall performance of the system.
- In case the query fails, the system starts its information extraction component to come up with a meaning for the unknown term. After a hypernym of a named entity such as "i-Punkt" has been extracted from the Internet and mapped to a concept of the ontology, the system can store the result as a new instance in the semantic memory.
- Using the knowledge that has been gained during the mapping process, the system can further identify semantic properties, which are then filled with additional data (e.g. *has-phone-number* etc.). For each semantic property the system tries to extract pieces of information.
- Apparently, the reliability of the extracted results varies, which is reflected in the semantic memory by an estimated confidence value for both instances and semantic properties.
- The semantic memory, as seen in the entity relationship diagram, has been structured in a way that for each instance an arbitrary number of properties can be held. This allows the system to introduce new properties and relations at run-time.