Ontology Learning in the Context of PortaGe

Pavel Smrž[1] Vít Nováček[2]

[1]Faculty of Information Technology,
Brno University of Technology, Czech Republic
E-mail: smrz@fit.vutbr.cz

[2]Faculty of Informatics,
Masaryk University, Brno Czech Republic
E-mail: xnovacek@fi.muni.cz

November 3, 2005

Outline

Outline

1. Introduction — PortaGe architecture

Outline

Outline

1. Introduction — PortaGe architecture
2. The role of ontologies in portal building
3. OLE — Ontology LEarning framework

Outline

1. Introduction — PortaGe architecture
2. The role of ontologies in portal building
3. OLE — Ontology LEarning framework
4. Preliminary results

Outline

1. Introduction — PortaGe architecture
2. The role of ontologies in portal building
3. OLE — Ontology LEarning framework
4. Preliminary results
5. Future directions

PortaGe — Basic Ideas

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data
- the target group – PhD students, young researchers

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data
- the target group – PhD students, young researchers
- long-term interest in the subject

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data
- the target group – PhD students, young researchers
- long-term interest in the subject
- an extension of Google Scholar and CiteSeer services

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data
- the target group – PhD students, young researchers
- long-term interest in the subject
- an extension of Google Scholar and CiteSeer services
- current search engines – keywords, phrases, document similarity

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data
- the target group – PhD students, young researchers
- long-term interest in the subject
- an extension of Google Scholar and CiteSeer services
- current search engines – keywords, phrases, document similarity
- digital libraries (ACM DL, Springer Link, arxiv.gov) – metainformation – author, journal, conference proceedings, year, . . .

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data
- the target group – PhD students, young researchers
- long-term interest in the subject
- an extension of Google Scholar and CiteSeer services
- current search engines – keywords, phrases, document similarity
- digital libraries (ACM DL, Springer Link, arxiv.gov) – metainformation – author, journal, conference proceedings, year, . . .
- results sorted according to relevance estimations

PortaGe — Basic Ideas

- the main aim – (semi)automatically generate a scientific web portal for a domain given by initial data
- the target group – PhD students, young researchers
- long-term interest in the subject
- an extension of Google Scholar and CiteSeer services
- current search engines – keywords, phrases, document similarity
- digital libraries (ACM DL, Springer Link, arxiv.gov) – metainformation – author, journal, conference proceedings, year, . . .
- results sorted according to relevance estimations
- what "relevant" means in each particular case

PortaGe — Initial Data

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field
2. seed documents and conference/project web pages relevant for the current search

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field
2. seed documents and conference/project web pages relevant for the current search
3. nodes in a current ontology (can be automatically extracted from the given and retrieved documents)

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field
2. seed documents and conference/project web pages relevant for the current search
3. nodes in a current ontology (can be automatically extracted from the given and retrieved documents)

PortaGe combines responses from several information sources:

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field
2. seed documents and conference/project web pages relevant for the current search
3. nodes in a current ontology (can be automatically extracted from the given and retrieved documents)

PortaGe combines responses from several information sources:

- search results from Google Scholar;

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field
2. seed documents and conference/project web pages relevant for the current search
3. nodes in a current ontology (can be automatically extracted from the given and retrieved documents)

PortaGe combines responses from several information sources:

- search results from Google Scholar;
- articles and papers found in digital libraries;

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field
2. seed documents and conference/project web pages relevant for the current search
3. nodes in a current ontology (can be automatically extracted from the given and retrieved documents)

PortaGe combines responses from several information sources:

- search results from Google Scholar;
- articles and papers found in digital libraries;
- information from freely accessible web services;

PortaGe — Initial Data

1. keywords, known authors, journals, conferences or projects characterizing the subject field
2. seed documents and conference/project web pages relevant for the current search
3. nodes in a current ontology (can be automatically extracted from the given and retrieved documents)

PortaGe combines responses from several information sources:

- search results from Google Scholar;
- articles and papers found in digital libraries;
- information from freely accessible web services;
- metainformation on hard-copies (books, journals, proceedings) in the faculty library and other traditional repositories.

PortaGe — Major Components

PortaGe — Major Components

- text mining for ontology acquisition;

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents
- citation analysis (provided by CiteSeer)

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents
- citation analysis (provided by CiteSeer)
- metasearch in digital libraries

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents
- citation analysis (provided by CiteSeer)
- metasearch in digital libraries
- analysis of “Publications” web pages

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents
- citation analysis (provided by CiteSeer)
- metasearch in digital libraries
- analysis of “Publications” web pages
- metadata annotation of web resources

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents
- citation analysis (provided by CiteSeer)
- metasearch in digital libraries
- analysis of “Publications” web pages
- metadata annotation of web resources
- merging of information

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents
- citation analysis (provided by CiteSeer)
- metasearch in digital libraries
- analysis of "Publications" web pages
- metadata annotation of web resources
- merging of information
- continuous search and source-change analysis

PortaGe — Major Components

- text mining for ontology acquisition;
- efficient local document classification and indexing;
- extraction of metainformation from the documents
- citation analysis (provided by CiteSeer)
- metasearch in digital libraries
- analysis of “Publications” web pages
- metadata annotation of web resources
- merging of information
- continuous search and source-change analysis
- portal personalization

The Role of Ontologies in PortaGe (1)

The Role of Ontologies in PortaGe (1)

The basic role consists in the definition of portal structures.

The Role of Ontologies in PortaGe (1)

The basic role consists in the definition of portal structures.

The core ontology contains concepts of publishers, books and book series, journals and their special issues, conferences, conference tracks workshops, projects, research teams, authors, papers, web pages, etc.

The Role of Ontologies in PortaGe (1)

The basic role consists in the definition of portal structures.

The core ontology contains concepts of publishers, books and book series, journals and their special issues, conferences, conference tracks workshops, projects, research teams, authors, papers, web pages, etc.

PortaGe supposes that the most of this can be shared among various scientific fields (different disciplines slightly differ in the conceptualisation of their research areas).

The Role of Ontologies in PortaGe (1)

The basic role consists in the definition of portal structures.

The core ontology contains concepts of publishers, books and book series, journals and their special issues, conferences, conference tracks workshops, projects, research teams, authors, papers, web pages, etc.

PortaGe supposes that the most of this can be shared among various scientific fields (different disciplines slightly differ in the conceptualisation of their research areas).

For a particular domain, it needs to be extended by individual instances of journals, conferences, etc.

The Role of Ontologies in PortaGe (1)

The basic role consists in the definition of portal structures.

The core ontology contains concepts of publishers, books and book series, journals and their special issues, conferences, conference tracks workshops, projects, research teams, authors, papers, web pages, etc.

PortaGe supposes that the most of this can be shared among various scientific fields (different disciplines slightly differ in the conceptualisation of their research areas).

For a particular domain, it needs to be extended by individual instances of journals, conferences, etc.

It is one of the tasks of the ontology extraction engine.

The Role of Ontologies in PortaGe (2)

The Role of Ontologies in PortaGe (2)

Ontologies used to classify the content of documents in PortaGe.

The Role of Ontologies in PortaGe (2)

Ontologies used to classify the content of documents in PortaGe.

Important especially for very narrow subfields with a limited number of documents that can be applied for training of the standard classifiers.

The Role of Ontologies in PortaGe (2)

Ontologies used to classify the content of documents in PortaGe.

Important especially for very narrow subfields with a limited number of documents that can be applied for training of the standard classifiers.

The automatic classification process can base its decision on the knowledge extracted from other documents in a previous run, such as the fact that a particular method is used for machine learning in other fields.

The Role of Ontologies in PortaGe (3)

The Role of Ontologies in PortaGe (3)

Ontologies provide mechanisms for context specification.

The Role of Ontologies in PortaGe (3)

Ontologies provide mechanisms for context specification.

Users can restrict the search for documents reflecting certain semantic relations based on the ontology, e.g. limit the output to the documents discussing “context-free grammars” as a “tool-for” “analysis of protein sequences”.

The Role of Ontologies in PortaGe (3)

Ontologies provide mechanisms for context specification.

Users can restrict the search for documents reflecting certain semantic relations based on the ontology, e.g. limit the output to the documents discussing "context-free grammars" as a "tool-for" "analysis of protein sequences".

The OLE framework interlinks individual pieces of such knowledge with lexico-syntactic patterns able to identify the relations in the retrieved documents.

The Role of Ontologies in PortaGe (4)

The Role of Ontologies in PortaGe (4)

Ontologies in personalization of multi-user portals.

The Role of Ontologies in PortaGe (4)

Ontologies in personalization of multi-user portals.

User profiles define rules to identify “the best” information for an individual user. A novice (in the given research domain) can ask for introductory documents, others prefer new information (the documents that appeared/were found in the last month), need a general summary of used methods (usually the most referenced documents), or focus on the relevance only.

The Role of Ontologies in PortaGe (4)

Ontologies in personalization of multi-user portals.

User profiles define rules to identify “the best” information for an individual user. A novice (in the given research domain) can ask for introductory documents, others prefer new information (the documents that appeared/were found in the last month), need a general summary of used methods (usually the most referenced documents), or focus on the relevance only.

The user profiles and the ontologies also cover the availability of the resources for a particular user, user-specified amount of documents that should be presented and processing time requirements.

Basic Requirements

Basic Requirements

- The process of ontology acquisition should run without any need of human assistance. On the other hand, the user must be able to influence the learning, refine the extracted, select relevant information and modify the stored data manually.

Basic Requirements

- The process of ontology acquisition should run without any need of human assistance. On the other hand, the user must be able to influence the learning, refine the extracted, select relevant information and modify the stored data manually.
- The amount of the processed resources can be very high (thousands of documents). The implementation of the ontology learning must be computationally efficient and robust.

Basic Requirements

- The process of ontology acquisition should run without any need of human assistance. On the other hand, the user must be able to influence the learning, refine the extracted, select relevant information and modify the stored data manually.
- The amount of the processed resources can be very high (thousands of documents). The implementation of the ontology learning must be computationally efficient and robust.
- The produced ontologies must reflect the stepwise development of the PortaGe system. If there is no current need for a particular kind of knowledge, the extraction should be postponed to later phases.

OLE Design

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;
- automatic inference of latent knowledge;

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;
- automatic inference of latent knowledge;
- QA interface for querying data stored in ontologies.

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;
- automatic inference of latent knowledge;
- QA interface for querying data stored in ontologies.

Core functionality:

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;
- automatic inference of latent knowledge;
- QA interface for querying data stored in ontologies.

Core functionality:

- extraction and efficient storage of domain concepts, concept clusters and their mutual relations;

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;
- automatic inference of latent knowledge;
- QA interface for querying data stored in ontologies.

Core functionality:

- extraction and efficient storage of domain concepts, concept clusters and their mutual relations;
- semantic searching and querying stored data;

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;
- automatic inference of latent knowledge;
- QA interface for querying data stored in ontologies.

Core functionality:

- extraction and efficient storage of domain concepts, concept clusters and their mutual relations;
- semantic searching and querying stored data;
- visualization of conceptual structures;

OLE Design

OLE (*Ontology LEarning*) system for knowledge acquisition and management addresses the following issues:

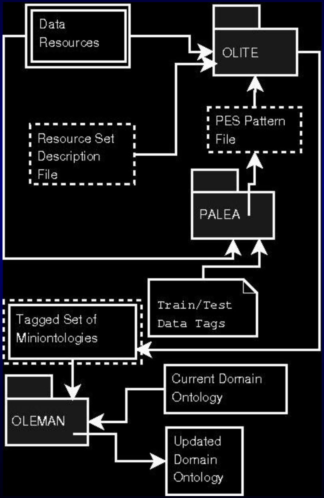
- iterative construction and maintenance of respective ontologies;
- explicit uncertainty representation;
- automatic inference of latent knowledge;
- QA interface for querying data stored in ontologies.

Core functionality:

- extraction and efficient storage of domain concepts, concept clusters and their mutual relations;
- semantic searching and querying stored data;
- visualization of conceptual structures;
- inference of implicit domain knowledge.

OLE Architecture (1)

OLE Architecture (1)



OLE Architecture (2)

OLE Architecture (2)

- OLITE processes plain or structured text and creates miniontologies.

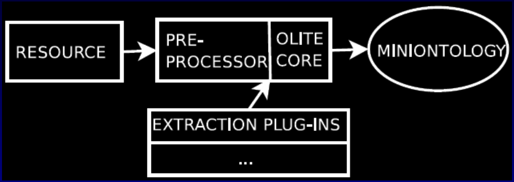
OLE Architecture (2)

- OLITE processes plain or structured text and creates miniontologies.
- PALEA is responsible for the learning of new semantic patterns. It is based on frequency-analysis and syntactic-pattern co-occurrence methods.

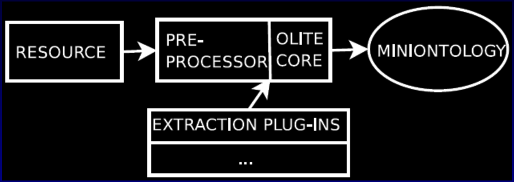
OLE Architecture (2)

- OLITE processes plain or structured text and creates miniontologies.
- PALEA is responsible for the learning of new semantic patterns. It is based on frequency-analysis and syntactic-pattern co-occurrence methods.
- OLEMAN merges the miniontologies and updates the base domain ontology. Uncertain information representation techniques are employed. The module can be used as a rudimentary ontology manager and question-answering system.

OLITE Work Flow

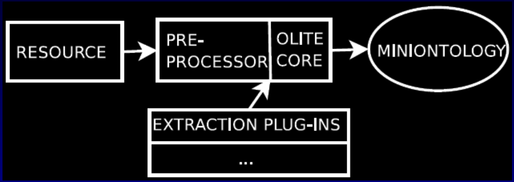


OLITE Work Flow



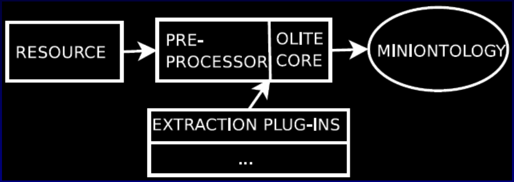
- **Resource** is a structured (XML, HTML) or unstructured (plain text) file

OLITE Work Flow



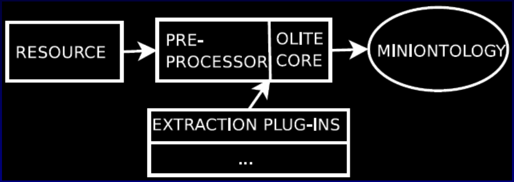
- **Resource** is a structured (XML, HTML) or unstructured (plain text) file
- **Preprocessor** incorporates generic NLP tasks such as tokenization, POS tagging and chunking.

OLITE Work Flow



- **Resource** is a structured (XML, HTML) or unstructured (plain text) file
- **Preprocessor** incorporates generic NLP tasks such as tokenization, POS tagging and chunking.
- **Extraction plug-ins** provide submodules implementing various extraction techniques

OLITE Work Flow



- **Resource** is a structured (XML, HTML) or unstructured (plain text) file
- **Preprocessor** incorporates generic NLP tasks such as tokenization, POS tagging and chunking.
- **Extraction plug-ins** provide submodules implementing various extraction techniques
- **Miniontology** covers the concepts and their relations identified in the respective resource.

OLITE Functional Components

OLITE Functional Components

- Tools supporting cross-language applicability:
  - resource reader interface
  - tagger traine
  - chunker trainer

OLITE Functional Components

- Tools supporting cross-language applicability:
  - resource reader interface
  - tagger traine
  - chunker trainer
- Preprocessor

OLITE Functional Components

- Tools supporting cross-language applicability:
    - resource reader interface
    - tagger traine
    - chunker trainer
- Preprocessor
- Language-specific analysis support

OLITE Functional Components

- Tools supporting cross-language applicability:
  - resource reader interface
  - tagger traine
  - chunker trainer
- Preprocessor
- Language-specific analysis support
- Extraction core with modular plug-in interface

OLITE Functional Components

- Tools supporting cross-language applicability:
  - resource reader interface
  - tagger traine
  - chunker trainer
- Preprocessor
- Language-specific analysis support
- Extraction core with modular plug-in interface
- Plug-ins of particular extraction methods

Cross-Language Applicability Tools

Cross-Language Applicability Tools

- Resource reader interface implements a set of transformations to convert the resource to the internal format.

Cross-Language Applicability Tools

- Resource reader interface implements a set of transformations to convert the resource to the internal format.
- Tagger trainer employs a tagged corpus to create a respective POS taggers (rule-based, stochastic, and Brill's transformation based).

Cross-Language Applicability Tools

- Resource reader interface implements a set of transformations to convert the resource to the internal format.
- Tagger trainer employs a tagged corpus to create a respective POS taggers (rule-based, stochastic, and Brill's transformation based).
- Chunker trainer employs a treebank-like corpus to learn how to chunk the input tagged sentences.

Preprocessing and Language-Specific Analysis Support

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

1. splitting the raw text into sentences;

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

1. splitting the raw text into sentences;
2. text tokenization;

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

1. splitting the raw text into sentences;
2. text tokenization;
3. POS tagging (Brill, stochastic, rule-based + unknown words);

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

1. splitting the raw text into sentences;
2. text tokenization;
3. POS tagging (Brill, stochastic, rule-based + unknown words);
4. chunking, esp. noun phrases (rule-based).

Language-Specific Analysis Support:

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

1. splitting the raw text into sentences;
2. text tokenization;
3. POS tagging (Brill, stochastic, rule-based + unknown words);
4. chunking, esp. noun phrases (rule-based).

Language-Specific Analysis Support:

- additional regular expressions for chunk parsing – keyword identification

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

1. splitting the raw text into sentences;
2. text tokenization;
3. POS tagging (Brill, stochastic, rule-based + unknown words);
4. chunking, esp. noun phrases (rule-based).

Language-Specific Analysis Support:

- additional regular expressions for chunk parsing – keyword identification
- terminological dictionaries

Preprocessing and Language-Specific Analysis Support

The preprocessing of the input goes through several phases:

1. splitting the raw text into sentences;
2. text tokenization;
3. POS tagging (Brill, stochastic, rule-based + unknown words);
4. chunking, esp. noun phrases (rule-based).

Language-Specific Analysis Support:

- additional regular expressions for chunk parsing – keyword identification
- terminological dictionaries
- WSD resources

Extraction Core

Extraction Core

1. Generic wrapper for chunked sentences
    - chunk splitting/merging
    - named-entity extraction
    - extraction of adjectival modifiers
    - predicate structures identification

Extraction Core

1. Generic wrapper for chunked sentences
   - chunk splitting/merging
   - named-entity extraction
   - extraction of adjectival modifiers
   - predicate structures identification
2. Interface for extraction plug-ins takes advantage of the wrapper methods and stores the extracted data in an internal ontology-representation format

Extraction Core

1. Generic wrapper for chunked sentences
   - chunk splitting/merging
   - named-entity extraction
   - extraction of adjectival modifiers
   - predicate structures identification
2. Interface for extraction plug-ins takes advantage of the wrapper methods and stores the extracted data in an internal ontology-representation format
3. Transformation layer provides transformational rules for immediate miniontology output in various formats (such as OWL or its probabilistic extension – BayesOWL); passes the unmodified extracted miniontology further to the integration module OLEMAN

Possible Extraction Methods

- pattern-driven extraction of semantic relations – well known and easy to implement method coined by Marti Hearst; utilizes matching of given patterns that are significant for particular semantic relations; mostly effective for the *is-a* relation but applicable for other semantic or ad hoc relations (such as *method-of* or *described-in* relations that are useful when analyzing scientific materials)

Possible Extraction Methods

- pattern-driven extraction of semantic relations – well known and easy to implement method coined by Marti Hearst; utilizes matching of given patterns that are significant for particular semantic relations; mostly effective for the *is-a* relation but applicable for other semantic or ad hoc relations (such as *method-of* or *described-in* relations that are useful when analyzing scientific materials)
- lexico-syntactic co-occurrence methods for clustering words , accompanied by identifying the classes using the knowledge already contained in our domain specific ontology (or external sources like WordNet, Roget's thesaurus, word sketch engines etc.);

Possible Extraction Methods

- pattern-driven extraction of semantic relations – well known and easy to implement method coined by Marti Hearst; utilizes matching of given patterns that are significant for particular semantic relations; mostly effective for the *is-a* relation but applicable for other semantic or ad hoc relations (such as *method-of* or *described-in* relations that are useful when analyzing scientific materials)
- lexico-syntactic co-occurrence methods for clustering words , accompanied by identifying the classes using the knowledge already contained in our domain specific ontology (or external sources like WordNet, Roget's thesaurus, word sketch engines etc.);
- various other kinds of semantic clustering or (F)FCA methods can be easily plugged in

Imprecise and Uncertain Information in Ontologies

Our ontology merging module works with an explicit uncertainty representation. It employs the fuzzy sets theory:

Imprecise and Uncertain Information in Ontologies

Our ontology merging module works with an explicit uncertainty representation. It employs the fuzzy sets theory:

- each relation R between concepts C and D in the resulting ontology has an appropriateness measure ($\mu_R(C, D)$ from the real interval $[0, 1]$) assigned – $\mu_R(C, D) = 1$ means that C and D are definitely in the R relation, $\mu_R(C, D) = 0$ means that C and D are definitely not in the R relation;

Imprecise and Uncertain Information in Ontologies

Our ontology merging module works with an explicit uncertainty representation. It employs the fuzzy sets theory:

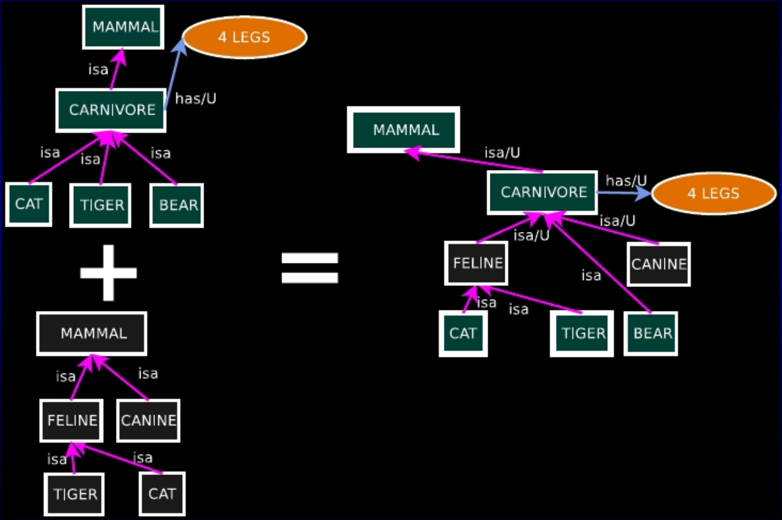
- each relation R between concepts C and D in the resulting ontology has an appropriateness measure ($\mu_R(C, D)$ from the real interval $[0, 1]$) assigned – $\mu_R(C, D) = 1$ means that C and D are definitely in the R relation, $\mu_R(C, D) = 0$ means that C and D are definitely not in the R relation;
- the measure is assigned to a relation instance according to the frequency of respective observations among the input data processed so far; so the number of occurrences of the concepts in a relation must be recorded when extracting the knowledge from data;

Imprecise and Uncertain Information in Ontologies

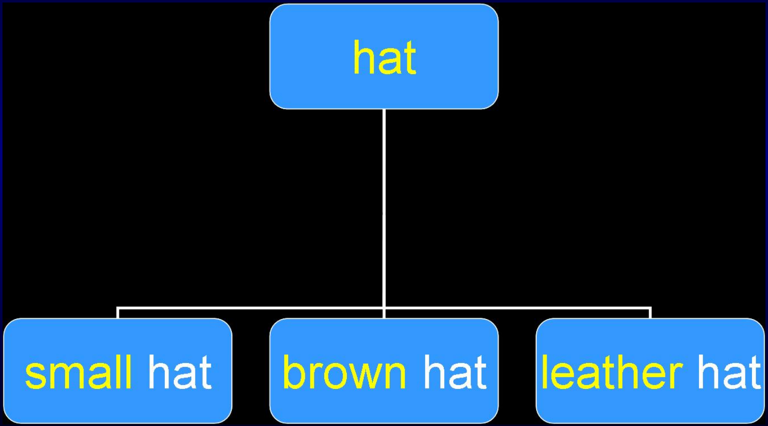
Our ontology merging module works with an explicit uncertainty representation. It employs the fuzzy sets theory:

- each relation R between concepts C and D in the resulting ontology has an appropriateness measure ($\mu_R(C, D)$ from the real interval $[0, 1]$) assigned – $\mu_R(C, D) = 1$ means that C and D are definitely in the R relation, $\mu_R(C, D) = 0$ means that C and D are definitely not in the R relation;
- the measure is assigned to a relation instance according to the frequency of respective observations among the input data processed so far; so the number of occurrences of the concepts in a relation must be recorded when extracting the knowledge from data;
- the initial appropriateness measure for concepts in miniontology is set to 1, but it can be modified when there is some vagueness indicator present in the concept context (for example in the sentence "Dogs have **usually** four legs.").

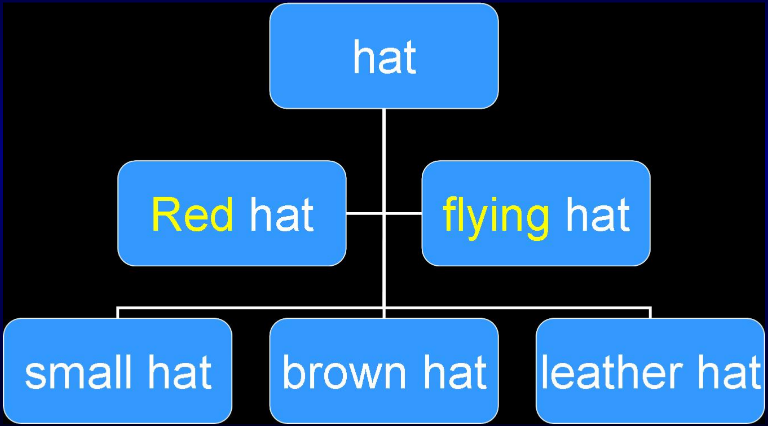
Ontology Merging



The Ontology of Hats



The Ontology of Hats



Preliminary Results

The method of pattern-based acquisition of simple relations was tested on English general corpus containing about 10^8 words in order to find out whether the proposed framework can provide enough data even when using simple extraction methods.

Selected *is-a* patterns	H_{abs}	H_{rel}	F_{all}	F_{acq}	$\frac{F_{acq}}{F_{all}}$
NP (and|or) other NP	17384	0.28	94	85	0.90
NP including (NPList (and|or))? NP	23985	0.38	92	73	0.79
NP (is|was) a NP	140632	2.26	66	30	0.45
(NPList)? NP like NP	147872	2.37	16	14	0.86
sums (H fields) and averages (F fields)	329873	5.29	67.00	50.50	0.75

- H_{abs} – numbers of matching sentences
- H_{rel} – relative frequency of matches
- F_{all} – ratio of successful pattern hits among randomly chosen sample of 50 matches
- F_{acq} – ratio of conceptual structures acquired by the OLITE module from the matches

Preliminary Results

type of the relation	subject	object
used_for	SCFG	RNA secondary structure prediction
described_in	CKY algorithm	Cocke-Kasami-Younger
is_a	ribosomal frameshifting	RNA function
abbr_means	HMM	Hidden Markov Models
abbr_means	SCFG	Stochastic Context-Free Grammars
is_a	RNA	molecule
is_a	protein	molecule

Conclusions and Future Directions

Conclusions and Future Directions

- more extraction plug-ins to increase the coverage of the OLITE module

Conclusions and Future Directions

- more extraction plug-ins to increase the coverage of the OLITE module
- defeasible mechanisms for ontology merging and their combination with fuzzy logic

Conclusions and Future Directions

- more extraction plug-ins to increase the coverage of the OLITE module
- defeasible mechanisms for ontology merging and their combination with fuzzy logic
- integration of advanced reasoning engines

Conclusions and Future Directions

- more extraction plug-ins to increase the coverage of the OLITE module
- defeasible mechanisms for ontology merging and their combination with fuzzy logic
- integration of advanced reasoning engines
- question-answering system for querying the ontologies

Conclusions and Future Directions

- more extraction plug-ins to increase the coverage of the OLITE module
- defeasible mechanisms for ontology merging and their combination with fuzzy logic
- integration of advanced reasoning engines
- question-answering system for querying the ontologies
- development of a framework for evaluation

Conclusions and Future Directions

- more extraction plug-ins to increase the coverage of the OLITE module
- defeasible mechanisms for ontology merging and their combination with fuzzy logic
- integration of advanced reasoning engines
- question-answering system for querying the ontologies
- development of a framework for evaluation
- WordNet, SUMO, MILO to define the directions for kinds of relations

Conclusions and Future Directions

- more extraction plug-ins to increase the coverage of the OLITE module
- defeasible mechanisms for ontology merging and their combination with fuzzy logic
- integration of advanced reasoning engines
- question-answering system for querying the ontologies
- development of a framework for evaluation
- WordNet, SUMO, MILO to define the directions for kinds of relations
- uncertainty via subjective language analysis