Master Thesis Presentation

Ontology Extraction for XML Classification

Student:

Natalia Kozlova

Supervisors:

Prof. Gerhard Weikum

Martin Theobald

Overview

- Introduction
- Problem description
  - Using ontology
  - Classifying XML
- Ontology creation
- Classification
- Conclusions and future work

Problem description

Classification using direct matching

- Lexical matching is loose in terms in capturing meaning
- Synonymy, polysemy and word usage pattern problems
- Fails with unknown words

Ontology can help

- Matching by sense, fighting synonymy, polysemy & …
- Stronger concepts, multi-word concepts allowed
- Possible to infer meaning of unknown concept
- Schema integration for XML classification
- No loss of precision with fewer training docs

Why not WordNet?

- WordNet usually offers much more then necessary
- WordNet is very broad, no topic specificity
- No weights

We want to get:

- More topic-specific ontology using complex concepts
  - can we generate reusable corpora-independent heuristics?
- Taxonomies from chosen strongly correlated parts of ontology
  - from small sets provided by user
- More precise document classification in the end

The role of XML

XML classification challenges

- Exploit annotation and structure
- Exploit ontological knowledge on sparse and/or heterogeneous training data
- Mapping of tags (and text terms) to semantic concepts

We want to get:

- Structural features

Typical XML structure

```
<computer>
  <notebook>
    <brand>Dell
      <monitor>15''</monitor>
      <ram>512</ram>…
    </brand>
    <brand>Sony
      <monitor>17''</monitor>
      <ram>512</ram>…
    </brand>
  </notebooks>
</computers>
```

Framework description

- Take corpora
- Create Ontology
  - Choose concepts
  - Extract relations (+ information from WordNet?)
  - Weight relations
  - Exploit user data to create Taxonomy
- Plug in classifier
- Classify new documents
  - Use XML structural features

Taxonomy example:

- Fine arts
- Mathematical and natural sciences
  - Astronomy
  - Biology
  - Computer science
    - Databases
    - Programming
    - Software engineering
  - Chemistry
  - …
- …

Overview

- Introduction
- Problem description
- Ontology creation
  - Corpora description
  - Concepts extraction
  - Relations extraction
  - Weights assignment
- Classification
- Conclusions and future work

Corpora description – Wikipedia (I)

- Wikipedia contains about 350000 articles
- Content is very broad; created by many authors
- Articles on many topics with different granularity levels
- Available in HTML and as database dump
- DB format is very convenient for experiments
- Internal markup is documented

- Drawbacks:
  - The structure is plain, no hierarchy
  - Multiple classification schemas exist
  - Made by human – errors are possible

Corpora description – Wikipedia (II)

- General topics ->subtopics
- Document doesn't know its broader topic
- Contain links to more narrow / related docs
- Important concepts are hyperlinked
  - Multi-word concepts!

Problems:

- Not all concepts selected
- Words ambiguity

Document

example

Mathematical and natural sciences ->Computer science

"In its most general sense, **computer science** (**CS**) is the study of computation and information processing, … Computer scientists study what programs can and cannot do (see computability and artificial intelligence), …"

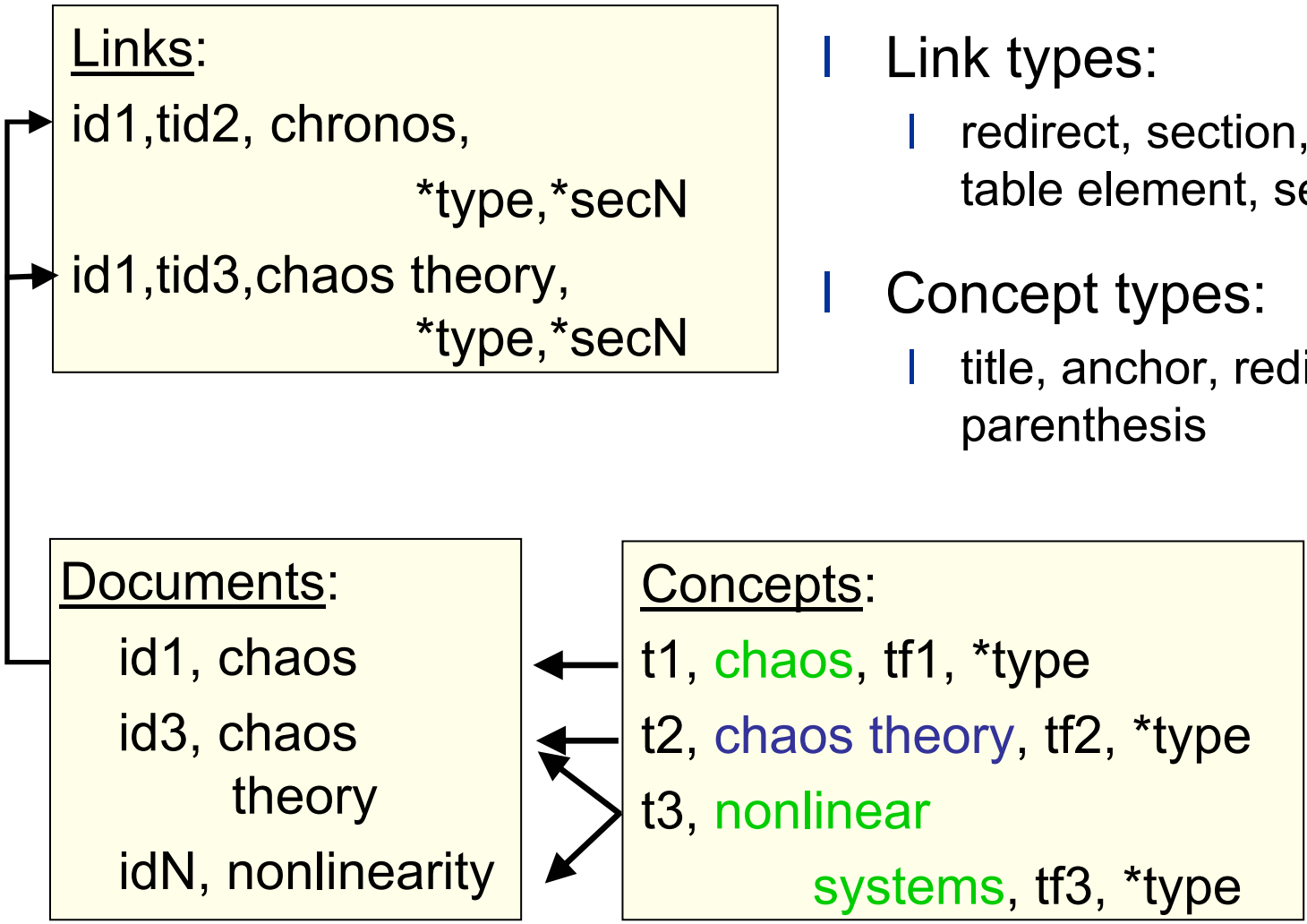
Ontology extraction given Wikipedia

- Study the structure of Wiki data, process as needed
- Create own data structures to store extracted information; create code for processing
- Parse (structure-aware) Wikipedia documents and extract links to another documents
- Find links between documents, select concepts and put them into ontology, count frequencies
- Investigate edge types between nodes in ontology
- Quantify edges

Concepts extraction and selection

- Wiki links contain a title of target document and possible "anchor"
  - [[America | United States]]; [[United States]]
- Concepts: titles => strongest, anchors => additional
- Additional sources of synonyms are "redirects"
  - Title: America; Content: [[#REDIRECT United States]]
- Can consider structural elements as
  - sections' headings; tables;
  - enumerations; in-doc positions; etc.
- Mark links, found in structural elements, accordingly

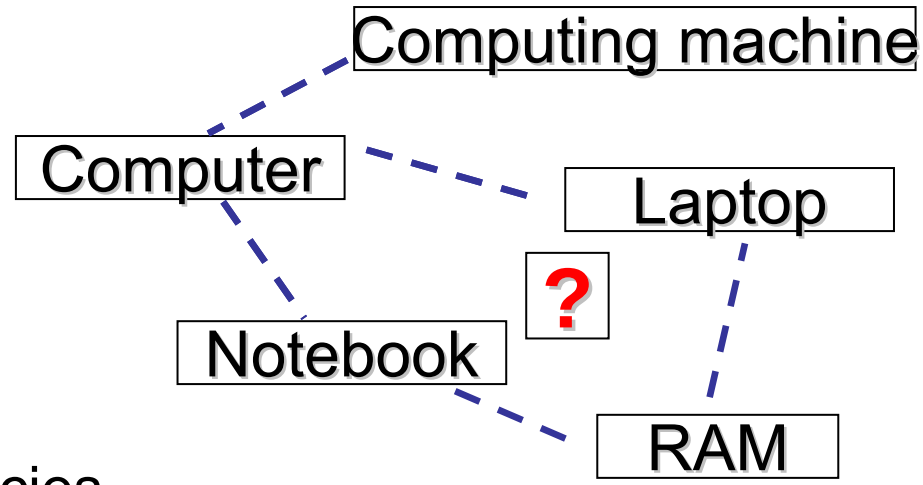
Links extraction



- Link types:
  - redirect, section, list element, table element, see also
- Concept types:
  - title, anchor, redirect, part in parenthesis

After concepts extraction

- At this point we have
  - Documents
  - Collections of doc's links
  - Concepts related to docs
  - Markup considered
  - Counted concepts' frequencies



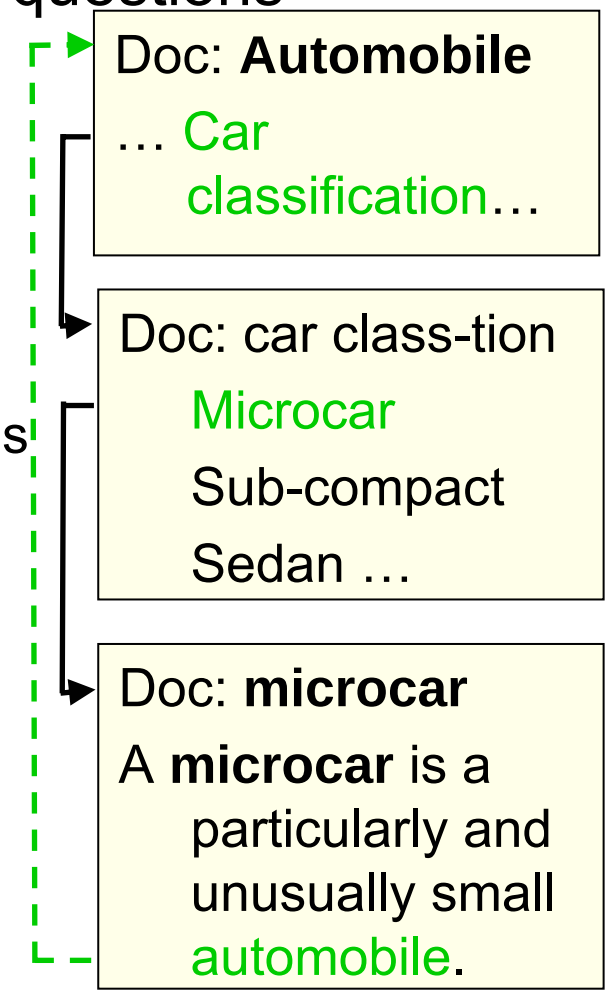
- Necessary to introduce
  - Edge types
    - Hypernyms (i.e. broader sense), hyponyms (i.e. kind of), meronyms (i.e. part of), …
  - Edge weights (measure of closeness between concepts)

What's next?

- Parse Wikipedia with concept phrases detection, store terms and links
  - For accuracy we can introduce geographical and person's names data
  - Compute term frequencies
- Generate a set of heuristics to reveal relationships (as independent as possible)
  - If "classification" is in the doc or section title and list is found, then the list elements are good candidates to be "is-a" related with the title concept
  - Links under "see also" are usually good candidates to be "kind-of" related with the title concept

Ontology creation: problems

- Edge types introducing – a lot of open questions
- Use parsing results and heuristics
- Consider simply relations first
  - Consider markup
  - Consider strong connections
- Some concepts point to many documents
  - It means that concept relation is ambiguous
- Use incremental mapping
  - Connect nodes step by step
  - Measure confidence
- Invoke WordNet in difficult cases
- Weight relations
  - Dice similarities?

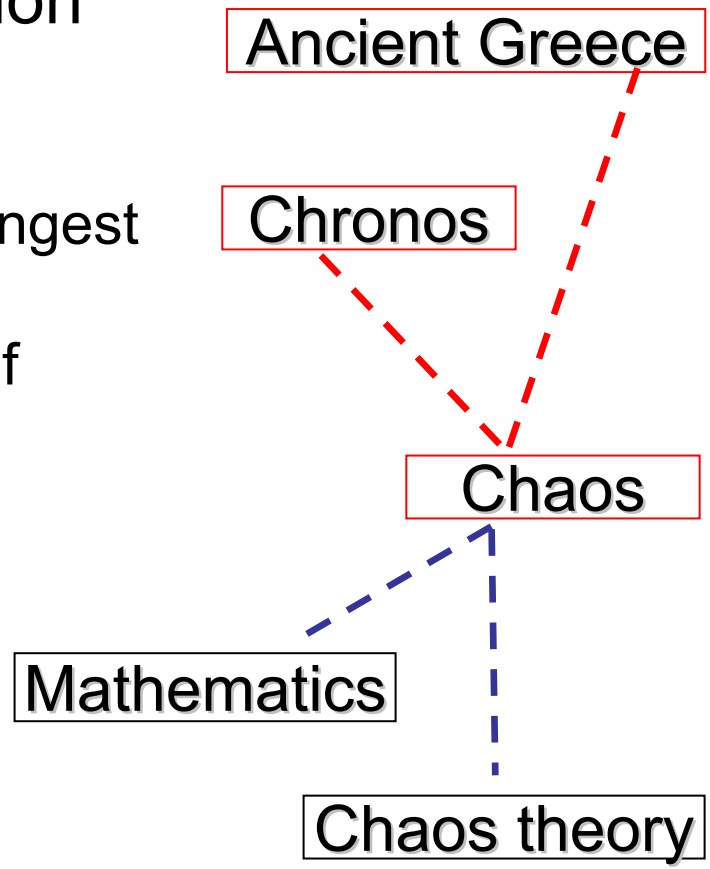


Overview

- Introduction
- Problem description
- Ontology creation
- Classification
  - Structure-aware document analyses
  - Mapping
- Conclusions and future work

Classification

- Taxonomy -target for classification
  - Obtain user information – what is needed
  - Select concepts that have the strongest correlation with provided data
  - Create a small “skeleton” subset of interest from ontology
  - Use this “part of interest”
- Parse new documents,
  - consider structure
- Classify
  - SVM, naïve bayes



Structure-aware document analyses

- Parse documents
  - Choose XML parser, modify if necessary
  - Consider annotations, possible attributes, tag-term pairs
  - Consider internal structure (twigs & tag paths)

Result: structural features

...
<computer>
<notebook>
<brand>Dell
<monitor>15”</monitor>
<ram>512</ram>…

Paths:

…computer-> notebook…

Twigs

notebook:

->ram

->monitor

Tags itself <…>

Mapping

- Map tags to senses
  - Take tag word(-s) and get sets of **senses** for them from ontology
  - Compare tag context *t* and term context *s* using cosine measure (i.e.)
  - Map tag to sense with highest similarity in context

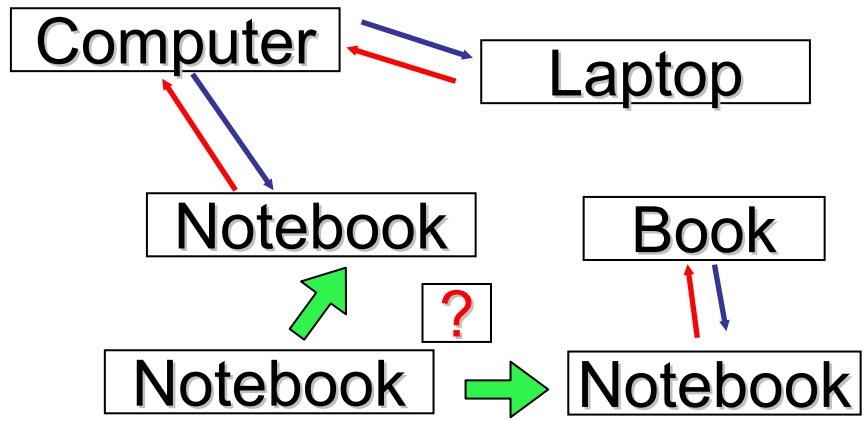
$$s' = \arg\max_{s'}(sim(con(t), con(s') \mid s' \in senses_{onto})$$

- Result: infer semantics from current context

<computer>
  <notebook>
    <brand>Dell
      <ram>512</ram>…

context(<tag>**)** =(text content (name, subordinate elements, their names))

context(term**)** =(hypernyms, hyponyms, meronyms, description)

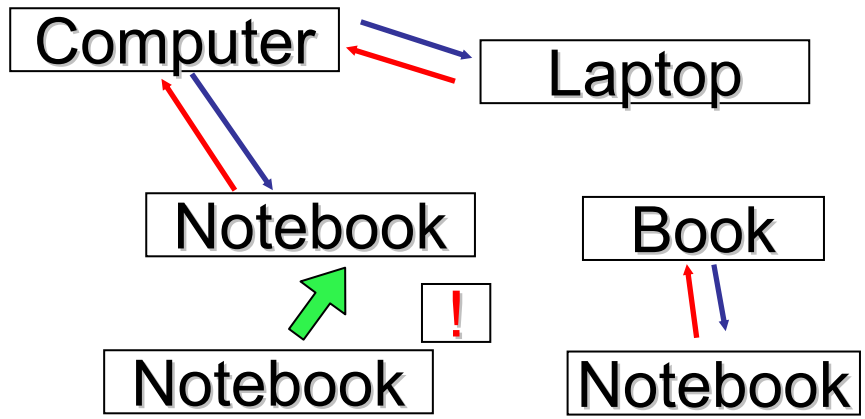


Classification summary

- Here we have:
- Ontology
  - the source of information to rely on
- Taxonomy
  - the target for classification
- Documents
  - represented as structural features
- Can classify

Taxonomy:

- Mathematical and natural sciences
  - Computer science
    - Programming
    - Software engineering



Conclusion

Ontology is better for:

- Matching by sense, fighting synonyms, polysemy problems
- Complex concepts; inferring meaning of unknown concept
- Processing different XML schemas

Concept-based classification boosts classification results

- Detection of synonyms
- Incremental mapping for unknown concepts

Structure-aware features offer a more precise representation for XML

Suggested framework is better for

- Training on small, user-specific topic directories
- Classification of heterogeneous data sources

Preliminary results and future work

- Done:
  - Document links parsed
  - Concepts obtained
  - Documents parsing in progress
- Future work
  - Work with WordNet
  - Finish ontological part
  - Force classification

The end

- Thank you for attention!
- Questions?