An adaptation of the Vector-Space Model for Ontology-Based Information Retrieval
Authors: Pablo Castells, Miriam Fernandez and David Vallet

Presented By: Charalampos Chelmis
Overview

- Approach
- Introduction
- Proposed System
- Experiments
- Discussion / Conclusion
Approach

• Model to improve search over large document repositories.
• Belief: a search engine must return documents.
• Approach:
  • *Ontology-based scheme for semi-automatic document annotation*
  • *Retrieval system*
    • *Adaptation of classic vector-space model*
    • *Annotation weighting algorithm*
    • *Ranking algorithm*
  • *Semantic search combined with conventional keyword-based*
• Experiments
  • *Scalability*
  • *Improvements*
Introduction

• **Semantic Search:**
  • *Motivation for Semantic Web*
  • *Ontologies vs. keyword-search.*

• **Semantic Search Engine:**
  • *Formal ontology-based queries*
  • *Knowledge base (KB)*
  • *Tuples of ontology values*

• **Typical use of Boolean search models**

• **Ideal view of the information space**
  • *Non-ambiguous,*
  • *Non-redundant,*
  • *Formal pieces of ontological knowledge.*
  • *Correct / incorrect answer*

• **Limitations**
  • *Cost of conversion*
  • *Document value ≠ sum of their pieces*
  • *For free text ontology values full-text search is needed*
  • *Scalability – ranking criteria*
Introduction (2)

- **Proposed System:**
  - *Ontology-based retrieval model*
  - *Exploit full-fledged domain ontologies and KBs*
  - *Support semantic search in DRs*

- **VS. Boolean Semantic Search Systems**
  - *Results are full documents*
  - *Considers both instance-level knowledge & topic taxonomies*
  - *Use of adapted VSM for scalability & ontology-based representation*
  - *Ranking algorithm on top*

- **Benefits**
  - *Inferencing capabilities*
  - *Interoperability bridge between heterogeneous systems*

- **Performance**

- **Limitations**
  - *Lack / incompleteness of available ontologies & KBs*
• Assumptions:
  • *Built KB is associated to DB.*
  • *No restrictions in domain ontology (some requirements)*
  • *Concepts/instances in KB are linked to document by means of explicit/non embedded annotations to the documents.*
  • *Reuse public KIM ontology & KB (extended manually)*
Semiautomatic Annotation

- DomainConcept instances use a label property to store the most usual text form of the concept class or instance.
- The property value can be set by an ontology designer or by semiautomatic means.
- The automatic concept → label mapping from KIM KB is used.
- Automatic annotator uses produced instance labels to find potential occurrences of instances in text documents.
- Whenever an instance label is found, an annotation is created between the instance and the document.
- Use of heuristics for polysemy:
  - The system always tries to find the longest label
  - Classification taxonomies are used as a source of semantic context for disambiguation
  - Short list of uncertain annotations are presented to domain expert
    - Unsolved polysemies
    - Indications that the right concept corresponding to the proper sense of a word is missing from the KB.
Weighting Annotations

- **Classic Vector Space model:**
  - *Keywords appearing in a document are assigned weights*

- **Proposed Model:**
  - *Annotations are weighted instead*

- **Weight computation:**
  - *Automatically*
  - *Adaptation of the TF-IDF algorithm*

\[
d_x = \frac{freq_{x,d}}{\max_y freq_{y,d}} \cdot \log \frac{|D|}{n_x}
\]

- \(freq_{x,d} \): # of occurrences in \(d\) of the keywords attached to \(x\)
- \(\max_y freq_{y,d} \): frequency of the most repeated instance in \(d\)
- \(n_x\): # of documents annotated with \(x\)
- \(D\): the set of all documents in the search space
Proposed System

- Query UI
  - RDQL Query
  - Query Engine
    - RDF KB
- Document Retriever
  - List of instances
  - Weighted annotation links
- Unordered Documents
- Ranked Documents
- Ranking
  - Document Base
• RDQL queries can express conditions involving:
  - *Ontology instances*
  - *Document properties*
  - *Classification values*

• As in classic keyword-based search SELECT variables can be weighted:
  - *Set manually*
  - *Automatically derived*

• Inferencing mechanisms are used for implicit query expansion
  - *Class hierarchies*
  - *Rules*
  - *KB is expanded by adding inferred statements beforehand*

• Set of tuples retrieved:
  - *Only domain concept instances*
  - *Document classes instances*
Ranking Algorithm

- Semantic similarity value between
  - Query
  - Each document
- O: the set of all classes & instances in the ontology
- D: the set of all documents
- q ∈ Q: an RDQL query
- V_q: the set of variables in the SELECT clause of q
- w: the weight vector for these variables (value range 0-1)
- T_q ⊂ O|V_q|: the list of tuples in the query result set
- document vector d ∈ D: representation of document in search space
- d_x: weight of the annotation of the document with concept x
- Extended query vector q_x
- Similarity between a document & a query: \[ \text{sim}(d, q) = \frac{d \cdot q}{|d| \cdot |q|} \]
• Keyword-based search will perform better when knowledge in KB is incomplete.

• **CombSUM strategy**
  - a method to combine the output of several search engines
  - Combined ranking score is linear combination
  - *Final score is* $\lambda \text{sim}(d,q) + (1-\lambda) \text{ksim}(d,q)$

• **Normalization step**
  - *Scale score to same range*
  - *Undo potential biases in the distribution of scores*

• **Keywords extraction**
  - *Automatically from the user query*
  - *From RDQL query*
Experiments

- 145,316 documents corpus from CNN Web site
- KIM domain ontology and KB
- Only 1 classification taxonomy
- Complete KB includes (compatible with RDF & OWL):
  - 281 classes
  - 138 properties
  - 35,689 instances
  - 465,848 sentences
  - 71MB in RDF format
- Automatic generation of concept-keyword mapping
  - 3 * 10^6 annotations
- Average observed response time below 30 sec
- Annotations are stored in separate db to avoid bottleneck
- Metrics based on manual ranking of all documents (0-5 scale)
- Weight of query variables set to 1
• Query:
  *News about banks that trade on NASDAQ, with fiscal net income greater than two million dollars*

• Keyword-based algorithm performs poorly
  *Limited expressive power*
  *Fails to express all the query conditions*

• KB contains many instances of banks

• News about matching banks are considered relevant

• Typical results when:
  *Search query involves ontology region with high degree of completeness*
  *KB doesn’t contain all banks*

• Since keyword-based result is poor, the linear combination value is also smaller.
Experiments (2)

- **Query:**
  
  *News about telecom companies*

- **KB contains only few instances**

- **Results:**
  
  - *Low precision for ontology-based approach*

  - *Since keyword-based result is better, the linear combination value is also better.*
Experiments (3)

- **Query:**
  News about insurance companies in USA.

- **Results:**
  - ontology-based approach fails
  - Performance is spoiled by incorrect annotations

- Since keyword-based result is better, the linear combination value is also better.
Experiments (4)

- Average performance comparison over 20 queries
- Results:
  - *Situations where ontology-only search performs bad are compensated on average*
• Performance comparison with conventional search systems
• Better recall:
  • *when querying for class instances,*
  • *by using class hierarchies & rules.*
• Better precision:
  • *by using structured semantic queries,*
  • *by using query weights,*
  • *by reducing polysemantic ambiguities.*
• Combination of conditions on concepts and contents
• Better results:
  • *With increase in the # of clauses in the formal query*
  • *With complete and high quality ontology / KB / concept labels*
• Further work needed:
  • *On automatic annotation techniques.*
  • *Weighting procedure.*
  • *Human supervision.*
  • *Score combination strategy.*
  • *Systematic efficiency testing.*
  • *Experimentation with heterogeneous data sets.*
  • *Model extension with profile of user interests for personalized search.*
Thank You