An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval

PABLO CASTELLS, MIRIAM FERNA´NDEZ, AND DAVID VALLET

PRESENTED BY : SUBODH SAH
Topics

- Aim
- Characteristics of the Model
- Knowledge Base And Document Base
- Weighting Annotations and Ranking Algorithm
- Processing Queries
- Experiments
- Improvements
- Conclusion
Vector Space Model

Q. What is Vector Space Model?

**Vector space model**: an algebraic model for representing text documents (and any objects, in general) as vectors of identifiers, such as, for example, index terms.

- information filtering
- information retrieval
- indexing and relevancy rankings.

Documents and queries are represented as vectors.

\[
d_j = (w_{1,j}, w_{2,j}, \ldots, w_{n,j})
\]

\[
q = (w_{1,q}, w_{2,q}, \ldots, w_{n,q})
\]
Aim

- Model for the exploitation of ontology-based knowledge bases to improve search over large document repositories.

- Ontology-based scheme
  - Semiautomatic annotation of documents
  - Retrieval system

- The retrieval model: adaptation of the classic vector-space model
  - Annotation weighting algorithm
  - Ranking algorithm
Performance and Incompleteness

Q. What is the performance of the model dependent on?

Ans. Direct relation with the

- Amount
- Quality of information within the KB it runs upon.
- Ontologies and Metadata (Semantic Web) is incomplete.
- Tolerance to incomplete KBs in this model.
- Combination with the keyword based search.
Characteristics of the Model

- **Conceptual Search** (search based on meaning rather than just character strings) – use detailed, densely populated ontology-based KB.

- **Flexibility of IR on unstructured search space** and the **expressiveness** and **detail** of a structured relational model.
  - But use Ontology based approach rather than conventional relational approach.

- **Similar to TAP and KIM** – automatic semantic annotation.
Knowledge Base And Document Base

Fig. 1. Root ontology classes.
Root Ontology Classes

- **Domain Concept** root of all domain classes.
- **Document** is used to create instances that act as proxies of documents from the information source to be searched upon.
  - Report, News, PurchaseOrder, Invoice, Message, etc.
  - Media Content – Extension for multimedia retrieval.
- **Topic** - root for class hierarchies used as classification schemes and are never instantiated. Taxonomies from as Open Directory Project (ODP) from WWW.
Document Annotation

- **Annotation ontology**: semantic indexing of documents with non-embedded annotations.

- **Annotations creation**:
  - manually by a domain expert (Manual Annotation subclass).
  - semi automatically (Automatic Annotation subclass).

- **Semiautomatic Annotation**: Domain Concept instances use a **label property** to store the most usual text form of the **concept class** or instance. Property: multi-valued (several textual lexical variants).

- Instance labels are used by the automatic annotator: find potential occurrences of instances in text documents and Mapping is created.
Polysemy

Q. What heuristics are used to solve the Polysemy problem?

Ans. 1) **Label coincidence** between different instances or classes. First, the system always tries to find the longest label, e.g., “Real Madrid” is preferred to “Madrid.”

2) **Classification taxonomies** are used as a source of semantic context for disambiguation: similarity measure.

For e.g., the word “Irises” in a document classified under Arts → Van Gogh’s famous Painting and not a subclass of Flower(provided painting instance in KB and classified as Arts).
Weighting Annotations and Ranking Algorithm

- In Classic Vector Space, **Keywords** were assigned **weights**.
- Here, **Annotations** are assigned a **weight**: relevancy of instance for the document meaning.
- Weights computed automatically by an adaptation of the TF-IDF algorithm, based on the frequency of occurrence of the instances in each document.

**Ranking**: Semantic similarity value between
  - Query
  - Each document
Processing Queries

- An ontology-based information retrieval system takes as input a formal RDQL query.
- The RDQL query is executed against the knowledge base, which returns a list of instance tuples that satisfy the query.
- Finally, the documents that are annotated with the instances returned in the previous step are retrieved, ranked, and presented to the user.

Fig. 2. Our view of ontology-based information retrieval.
The RDQL queries can express conditions:
- domain ontology instances
- document properties (such as author, date, publisher, etc.)
- classification values.

i.e., “cultural articles published by the Le Monde newspaper about European movies with Canadian actors in the cast.

SELECT clause of the RDQL query - variables weighted( relative interest of the user for each of variable from document).
Our system uses inferencing mechanisms for implicit query expansion based on class hierarchies.

In order to illustrate our model, consider the query “Players from USA playing in basketball teams of Catalonia.” This would be formalized as:

```
SELECT ?player ?team
WHERE
  (?player <rdf:type> <kb:SportsPlayer>)
  (?player <kb:plays> <kb:Basketball>)
  (?player <kb:nationality> <kb:USA>)
  (?player <kb:playsIn> ?team)
  (?team <kb:locatedIn> <kb:Catalonia>)
```
Experiments

Query a. “News about banks that trade on NASDAK, with fiscal net income greater than two billion dollars.”

[In this example, the semantic retrieval algorithm outperforms keyword-based search because the limited expressive power of the latter fails to express all the conditions in the query.]

Query b. “News about telecom companies.”

[In this example, the ontology KB has only a few instances of telecom companies, so not all documents relevant to the query are annotated.]

Query c. “News about insurance companies in USA.”

[This example shows a case where our method fails. The performance of the semantic retrieval is spoiled by incorrect annotations, namely, the “Kaye” insurance company is confused with “Kaye” as a person’s name.]
Fig. 4. Evaluation of ontology-based search (alone and combined with keyword-based) against keyword-based only. The performance of the algorithms, in terms of precision versus recall figures (as defined in, e.g., [31]), is shown for three different queries (a), (b), and (c), and averaged over 20 queries (d).
Discussion

Performance comparison with conventional search systems

Figure 5. Comparative precision histogram for semantic search versus keyword-based search.
Improvements

- **Better recall** when querying for class instances.

- **Better precision** by using *structured semantic queries*. Structured queries allow expressing more precise information needs, leading to more accurate answers.

- **Better precision** by reducing *polysemic ambiguities* using instance labels and classifications of concepts and documents.

- **Better results** with use of high quality ontology/KB.
Conclusion and Future Work

- Continuation of previous work on the construction, exploitation, and maintenance of domain-specific KBs using Semantic Web technologies.

- Achieve better levels of knowledge coverage by formal KBs - *(semi)automatic integration* of *independently developed* and maintained KBs.

- Improvement in *annotation weighting scheme*. Take advantage of structured document – Title,Body.

- *Annotating documents with statements*, besides instances, is another interesting possibility to experiment with.
Bonus!!!(How google is doing it?)

Q. What is the Google Knowledge Graph?

http://www.searchenginejournal.com/google-killing-branded-search-traffic-mobile-phones-heres-fix-issue/123551/
Thank You  & Be Happy!!! 😊