Semantic Search in Documents Enriched by LOD-based Annotations

Pavel Smrz and Jan Kouril
Brno University of Technology
Faculty of Information Technology
IT4Innovations Centre of Excellence
Bozetechova 2, 612 66 Brno, Czech Republic
{smrz,ikouril}@fit.vutbr.cz

Abstract

This paper deals with information retrieval on semantically enriched web-scale document collections. It particularly focuses on web-crawled content in which mentions of entities appearing in Freebase, DBpedia and other Linked Open Data resources have been identified. A special attention is paid to indexing structures and advanced query mechanisms that have been employed into a new semantic retrieval system. Scalability features are discussed together with performance statistics and results of experimental evaluation of presented approaches. Examples given to demonstrate key features of the developed solution correspond to the cultural heritage domain in which the results of our work have been primarily applied.

Keywords: semantic search, text annotation, semantic enrichment

1. Introduction

Semantic enrichment of textual content and information retrieval based on it has become a popular research topic recently (Cunningham et al., 2011; Suchanek et al., 2013; Chakrabarti et al., 2012; Li et al., 2010). Having entity annotations produced by services such as DBpedia Spotlight[1] or Lupedia[2], resulting texts need to be indexed in an appropriate way to support advanced semantic search mechanisms, similarity search based on entity co-occurrence, etc. This paper tackles the topic of indexing semantically annotated texts by comparing two alternative indexing approaches.

To understand decisions made in the presented work, it is crucial to distinguish two opposite views on the semantic search on textual resources. Individual entities as well as basic relations among them can be automatically extracted from the text and stored in a knowledge base. In many contexts, it is fully sufficient to query just the resulting knowledge base (e.g., via SPARQL over an RDF store) without any need for dealing with the source texts. On the other hand, knowledge structures that would define an extraction task can be perceived as not clear or stable enough to rely on in other contexts. Then, the concept of fulltext search on semantically enhanced content is generally preferred. For example, one can search for Europeana records mentioning an Irish architect together with feminism (within a sentence). This would take form of the following query in our system (see more information on the query language in Section 3):

```
feminis* near mentions.person
              .nationality:Irish
              .field: architecture;
dataset: Europeana
```

We focus on this kind of searches in the presented work.

To be feasible, indexing of the semantically enriched content assumes that there is a limited number of attributes for each identified entity. Often, there is only a detailed type (such as *sculptor* or *lake*) to be stored. In any case, the complete set of attributes is expected to be known in the indexing time – it is either directly provided by an external knowledge base; or it is inferred with a help of a reasoning engine and materialized in advance.

The context in which the implemented semantic indexing is primarily used is defined by the DECIPHER project. It aims to support the discovery and exploration of cultural heritage through story and narrative. Stories are entered by museum professionals and used in the whole range of narrative construction and knowledge visualisation. To be able to do this, the system needs to "understand" the content – to transform free-text narratives into a structured form with explicitly identified semantics. Advanced search interfaces employ automatically extracted metadata anchored in the LOD resources (mainly Freebase and DBpedia) and identified entities and relations also help to quantify semantic relatedness of stories in the similarity search.

2. Indexing annotation data

ElasticSearch[3] – an indexing engine based on the Apache Lucene[4] – forms the core of our indexer. The tool communicates through a JSON-based API[5]. It is implemented in Java but various bindings for other programming languages such as Python (PyEs[6]) or PHP (Elastica[7]) can be used too. The processing of web-scale data benefits from inherent distributional character of the schema-free multitenancy engine as scalability and high-availability characteristics are available "out of box".

Advanced annotation indexing mechanisms discussed in the following paragraphs were implemented as specific plugins on the level of the Lucene library. Two alternative approaches were explored. The first one implements a compressed annotation index scheme inspired by the Snippet

[1] `http://spotlight.dbpedia.org/?`
[2] `http://lupedia.ontotext.com/?`
[3] `http://www.elasticsearch.org/`
[4] `http://lucene.apache.org/`
[5] `http://jsonapi.org/`
[6] `http://pyes.readthedocs.org`
[7] `http://elastica.io/`

Interleaved Posting coding (SIP) introduced in (Chakrabarti et al., 2012). It constructs separate inverted indices for entities and for annotations. The posting list of an annotation inverted index records token spans of entity mentions in particular documents. A special compressing is employed – span offsets are gap-gamma coded and entity IDs are shortened by a per-document dictionary mapping (see (Chakrabarti et al., 2012) for details on the compressing structures).
The second annotation indexing approach builds on the idea of placing multiple tokens on the same position. ElasticSearch does not directly support such a scheme but the concept of synonyms can be utilised for this purpose. A special plugin – a custom token analyser – was implemented that transforms a text with annotations to a form interpreted as a sequence of words with identified "synonyms". Metadata is then placed "within" a given token, which ensures correctness of proximity querying. The following lines give an example of the input to the custom token analyser:

```
Vouet[person;painter_role;
      craftsman_role;
      French_nationality;
      European_nationality]
visited[activity;travel]
Rome[location;Italy_country;
     Europe_continent]
```

The metadata appears right behind recognized entities, within brackets, and the custom token analyser recognizes them as synonyms indexed at the same offset.
In contrast to standard Lucene indices, our plugin enables enriching every single token by an annotation. Yet, it has an advantage over other indexing schemata for metainformation, such as that used in the MG4J system[8] (Boldi and Vigna, 2005). It allows adding arbitrary numbers of annotation attributes to individual tokens. Moreover, one attribute can have multiple values and there is no need to define an upper limit of their number. This is also beneficial for updating the index by new annotations – there is no need to re-index the entire corpus.
Queries referring to a single annotation attribute for an individual position are processed as standard keyword queries. Special Lucene constructs – span queries – are employed to deal with multi-attribute queries:

```
a=SpanTermQuery("doc","painter_role")
b=SpanTermQuery("doc","16_century")
f=SpanNearQuery(clauses=[a,b],slop=0)
```

The whole document indexing follows a standard procedure – semi-structured documents are semantically enriched, transformed into the JSON format and stored into the main index. Type-specific metadata from the input (such as a specific field identified in the source) is stored in separate fields that can be easily queried. The metadata added by the annotation process corresponding to entities and relations identified in the text are transformed to a form appropriate for the first or the second indexing approach and the final index is created.

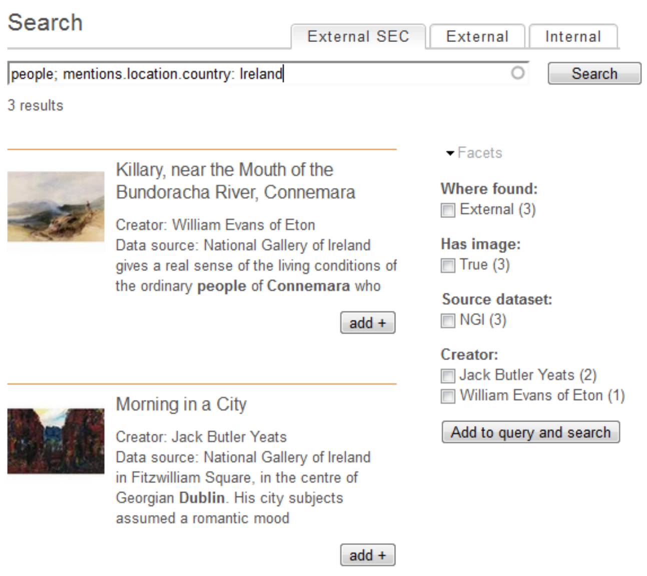
[8]http://mg4j.di.unimi.it/

3. Semantic querying

Users can search relevant indexed fields by simple fulltext queries – entering individual words, phrases, etc. Advanced queries are formed as a combination of keyword queries and a set of specific field queries. Semicolons are used as separators.
Keyword `mentions` is used to query entities and their attributes identified by the annotation process. In the case of hierarchically organized concepts (such as cities belonging to countries that further belong to continents), any mention of an entity on a lower level is counted as a mention of all entities higher in the hierarchy too. For example, the query:

```
mentions.location: Britain
```

returns all documents mentioning not only the United Kingdom, Great Britain or the U.K. (if it is disambiguated as referring to the particular location), but also all locations known to be in the U.K. Figure 1 demonstrates such a query.


Search
External SEC | External | Internal
people; mentions.location.country: Ireland
Search
3 results

Killary, near the Mouth of the Bundoracha River, Connemara
Creator: William Evans of Eton
Data source: National Gallery of Ireland
gives a real sense of the living conditions of the ordinary **people** of **Connemara** who
add +

Morning in a City
Creator: Jack Butler Yeats
Data source: National Gallery of Ireland
in Fitzwilliam Square, in the centre of Georgian **Dublin**. His city subjects assumed a romantic mood
add +

Facets
Where found:
External (3)
Has image:
True (3)
Source dataset:
NGI (3)
Creator:
Jack Butler Yeats (2)
William Evans of Eton (1)
Add to query and search

Figure 1: A combination of a keyword-based search and a query on the semantically enriched content

Inferred knowledge (corresponding to entity attributes stored in the index) can be queried in a similar way. For example, the query:

```
mentions.person.nationality: Irish
```

finds documents mentioning any known (art-related) person whose nationality is Irish.
The semantic enrichment is also applied to specific fields so that it is possible to formulate advanced queries such as:

```
creator.influenced_by: Pablo Picasso;
genre: portrait
```

which would search for portraits by artists influenced by Pablo Picasso.

4. Experimental evaluation

Five datasets were used in our experiments – ClueWeb12[9] and CommonCrawl[10] annotated by means of the DECIPHER Semantic Annotator (Smrz et al., 2013), FACC1 – Freebase Annotations of the ClueWeb Corpora (Gabrilovich et al., 2013) and the Wikilinks corpus (Singh et al., 2012) provided by Google, and a corpus formed by material collected within the DECIPHER project (including complete Europeana[11]).

Indexing used Elastic search version 0.90 and runs on 20 nodes, each with Xeon E5-2630 (Sandy Bridge-EP), 6/12 cores, 2.3-2.8 GHz, 15 MB cache, 32 GB DDR3-1333 RAM and three 3 TB disks in RAID for data.

The two indexing approaches discussed in Section 2 were compared in terms of the indexing time, the overall size of resulting index structures and an average querying time on two evaluation sets of queries. The first set is formed by typical TREC entity queries[12]. The second one was designed to reflect querying needs of the advanced search in DECIPHER. It is based on the statistics from the Google FACC1 corpora reflecting the power law of entity occurrences in web-crawled data (Gabrilovich et al., 2013).

The pre-processing and indexing time corresponded to about 47,000 tokens per second. The difference between the two approaches in the indexing time was not statistically significant – it was dominated by the stream processing that merges data from all nodes participating in the pre-processing pipeline. However, the two approaches differed significantly in term of the index size and the query-processing time. The graph in Figure 2 shows how the index size depends on the "annotation overhead" – the ratio of the total number of features (types of entities identified in the text and all their attributes) to tokens. The size of the indexing structures for the synonyms-based method is generally smaller than in the case of the SIP method.

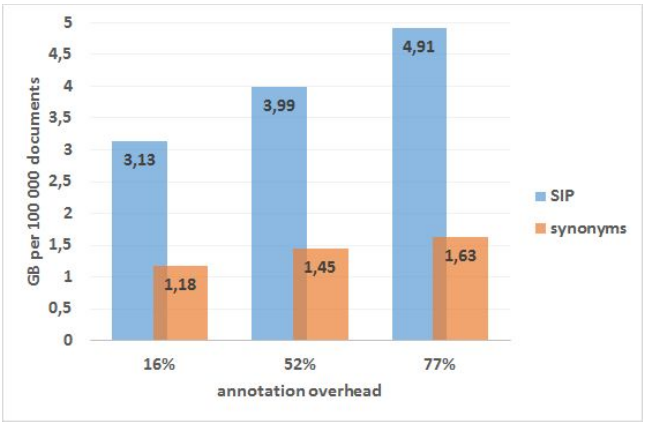


Figure 2: Comparison of index sizes of the SIP and synonyms-based indexing approaches

This finding can be related to the comparison of average response time measured on a set of 1000 queries reflecting real-user interaction with the semantic indexing system. The synonyms-based approach enables about 3 times faster query evaluation than the SIP method. The average time corresponds to 42 ms per query for the former and 124 ms for the latter.

The SIP modules implemented by original authors of the method are not production-ready; their parallel Hadoop-based versions are being prepared. To be able to deal with the full range of complex semantic queries, further experiments compared the developed solution with the MG4J system discussed in the previous section and the Manatee corpus manager (Rychly, 2007) that can cope with very large annotated data. MG4J and Manatee were run on the same source data with annotations in separate columns (attributes). Note, that Manatee enables indexing multiple values for a single attribute but this feature was not used in our experiments.

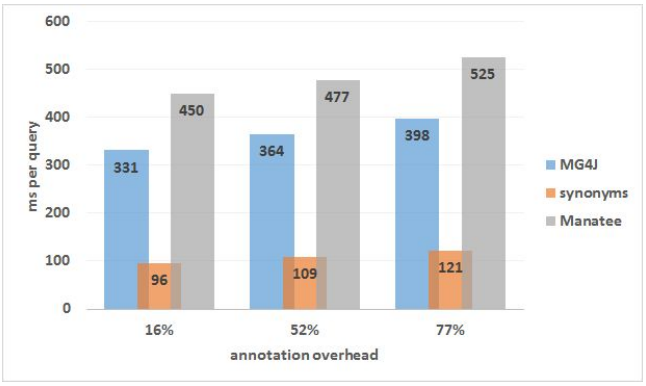


Figure 3: Average query times of compared systems

Figure 3 compares average query times of the three systems. Our solution remains to be 3–4 times faster than the other two; independently of the amount of metadata indexed. The synonyms-based approach seems to be optimal for the kind of semantic queries explored in our research.

5. Conclusions

The semantic search solution introduced in this paper demonstrates advanced features of the fulltext search in the semantically-enriched content. A novel indexing schema was compared to other state-of-the-art approaches and showed significant improvements over those systems in real-world experiments. The realized system which takes advantage of the described indexing technique was successfully applied in the DECIPHER project and integrated to the end-user tool that museum professionals interact with.

6. Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 2700001 (project DECIPHER – Digital Environment for Cultural Interfaces; Promoting Heritage, Education and Research) and by the IT4Innovations Centre of Excellence project, Reg. No. CZ.1.05/1.1.00/02.0070, supported by Operational Programme Research and Development for

[9]http://boston.lti.cs.cmu.edu/clueweb12/
[10]http://commoncrawl.org
[11]http://www.europeana.eu/
[12]http://trec.nist.gov/data/entity.html

Innovations funded by Structural Funds of the European Union and the state budget of the Czech Republic.

7. References

Boldi, P. and Vigna, S. (2005). MG4J at TREC 2005. In Voorhees, E. M. and Buckland, L. P., editors, *The Fourteenth Text REtrieval Conference (TREC 2005) Proceedings*, number SP 500-266 in Special Publications. NIST. `http://mg4j.di.unimi.it/`.

Chakrabarti, S., Kasturi, S., Balakrishnan, B., Ramakrishnan, G., and Saraf, R. (2012). Compressed data structures for annotated web search. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 121–130, New York, NY, USA. ACM.

Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M. A., Saggion, H., Petrak, J., Li, Y., and Peters, W. (2011). *Text Processing with GATE (Version 6)*.

Gabrilovich, E., Ringgaard, M., and Subramanya, A. (2013). Facc1: Freebase annotation of clueweb corpora. Version 1 (Release date 2013-06-26, Format version 1, Correction level 0), June 2013, http://lemurproject.org/clueweb12/FACC1/.

Li, X., Li, C., and Yu, C. (2010). Entityengine: answering entity-relationship queries using shallow semantics. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, CIKM '10, pages 1925–1926, New York, NY, USA. ACM.

Rychly, P. (2007). Manatee/bonito – a modular corpus manager. In *1st Workshop on Recent Advances in Slavonic Natural Language Processing*, pages 65–70. Masaryk University.

Singh, S., Subramanya, A., Pereira, F., and McCallum, A. (2012). Wikilinks: A large-scale cross-document coreference corpus labeled via links to Wikipedia. Technical Report UM-CS-2012-015, University of Massachusetts, Amherst. `http://googleresearch.blogspot.cz/2013/03/learning-from-big-data-40-million.html`.

Smrz, P., Otrusina, L., Kouril, J., and Dytrych, J. (2013). Decipher deliverable d4.3.1: Semantic annotator. `http://decipher-research.eu/deliverables-resources`.

Suchanek, F., Fan, J., Hoffmann, R., Riedel, S., and Talukdar, P. P. (2013). Advances in automated knowledge base construction. In *SIGMOD Records journal*, March.