ARS/SD: An Associative Retrieval Service for the Semantic Desktop

Peter Scheir, Chiara Ghidini, Roman Kern, Michael Granitzer, and Stefanie N. Lindstaedt

Abstract. While it is agreed that semantic enrichment of resources would lead to better search results, at present the low coverage of resources on the web with semantic information presents a major hurdle in realizing the vision of search on the Semantic Web. To address this problem we investigate how to improve retrieval performance in a setting where resources are sparsely annotated with semantic information. We suggest employing techniques from associative information retrieval to find relevant material, which was not originally annotated with the concepts used in a query. We present an associative retrieval service for the Semantic Desktop and evaluate if the use of associative retrieval techniques increases retrieval performance.

Evaluation of new retrieval paradigms, as retrieval in the Semantic Web or on the Semantic Desktop, presents an additional challenge as no off-the-shelf test corpora for evaluation exist. Hence we give a detailed description of the

Peter Scheir
Graz University of Technology, Austria
e-mail: `peter.scheir@tugraz.at`

Chiara Ghidini
Fondazione Bruno Kessler, Italy
e-mail: `ghidini@fbk.eu`

Roman Kern
Know-Center, Inffeldgasse 21a, 8010 Graz, Austria
e-mail: `rkern@know-center.at`

Michael Granitzer
Know-Center, Inffeldgasse 21a, 8010 Graz, Austria
e-mail: `mgrani@know-center.at`

Stefanie Lindstaedt
Know-Center, Inffeldgasse 21a, 8010 Graz, Austria
e-mail: `slind@know-center.at`

S. Schaffert et al. (Eds.): Networked Knowledge - Networked Media, SCI 221, pp. 95–111.
springerlink.com

approach taken to the evaluation of the information retrieval service we have built for the Semantic Desktop.

1 Introduction

It is largely agreed that the semantic enrichment of resources provides for more information to be used during search (see e.g. [12] or [26]). In turn, this can lead to greatly improve the effectiveness of retrieval systems, not only for resources on the web but also for personal desktops. However, critics [17] as well as advocates [21] of the Semantic Web agree that only a small fraction of resources on the current web are enriched with semantic information. The sparse annotation of resources with semantic information presents a major obstacle in realizing search applications for the Semantic Web or the Semantic Desktop, which operate on semantically enriched resources. To overcome this problem, we propose the use of techniques from associative information retrieval in order to find relevant resources, even if no semantic information is provided for those resources.

The main idea of our approach is to perform search using spreading activation in a two layer network structure (graphically illustrated in Figure 1) which consists of (1) a layer of concepts, used to semantically annotate a pool of resources, and (2) a layer of resources (documents). The combination of spreading activation in both layers, traditionally performed either to find similar concepts or to find similar text, allows extending search to a wider network of concepts and resources, which can lead to the retrieval of relevant resources with no annotation.

In this paper we describe our approach towards information retrieval on the Semantic Desktop and present a retrieval service developed during the first year of the APOSDLE[1] project. The rest of this paper is organized as follows: in section 2 we introduce the concept of the Semantic Desktop and of associative information retrieval. In section 3 we describe the approach taken to the realization of the retrieval service. In section 4 we present the setting (APOSDLE) in which the retrieval service for the Semantic Desktop was employed and in section 5 we focus on the evaluation of the retrieval service. We present related work in section 6 and our conclusion in section 7.

2 Basic Concepts

The work presented in this paper provides a first implementation of an associative retrieval service for the Semantic Desktop. In this section we briefly introduce the main ideas and goals of the Semantic Desktop and of associative information retrieval.

[1] `http://www.aposdle.org/` (14.04.2008)

2.1 Semantic Desktop

The Semantic Desktop [24] [9] paradigm stems from the Semantic Web movement and aims at applying technologies developed for the Semantic Web to desktop computing. In recent years the Semantic Web movement led to the development of new, standardized forms of knowledge representation and technologies for coping with them such as ontology editors, triple stores or query languages. The Semantic Desktop founds on this set of technologies and introduces them to the desktop to ultimately provide for a closer integration between (semantic) web and (semantic) desktop.

2.2 Associative Information Retrieval

Crestani [8] understands *associative retrieval* as a form of information retrieval which tries to find relevant information by retrieving information that is by some means *associated* with information that is already known to be relevant. Information items which are associated can be documents, parts of documents, extracted terms, concepts, etc. The idea of associative retrieval dates back to the 1960s, when researches [22], [23] in the field of information retrieval tried to increase retrieval performance using associations between documents or index terms, which were determined in advance.

Association of information is frequently modeled as graph, which is referred to as *associative network* [8]. Nodes in this network represent information items such as documents, terms or concepts. Edges represent associations between information items and can be weighted and / or labeled, expressing the degree and type of association between two information items, respectively.

3 An Associative Information Retrieval Service for the Semantic Desktop

The service presented here relies upon the existence of two sources of information: first a domain ontology, used to define the vocabulary (concepts) used to annotate resources, and then the resources themselves in the form of textual documents. On top of these two sources of information we build an associative network consisting of two interconnected layers, one for concepts and one for documents (see Figure 1).

Nodes in the concepts layer correspond to concepts in the domain ontology. Nodes in the document layer correspond to documents on the Semantic Desktop. Concept nodes are associated by means of semantic similarity (cf. section 3.1), while document nodes are associated by means of textual similarity (cf. section 3.2). The link between the two layers of the network is provided by annotations: a concept node is associated with a document node if the concept is used to annotate that document (cf. sections 3.3 and 3.4).

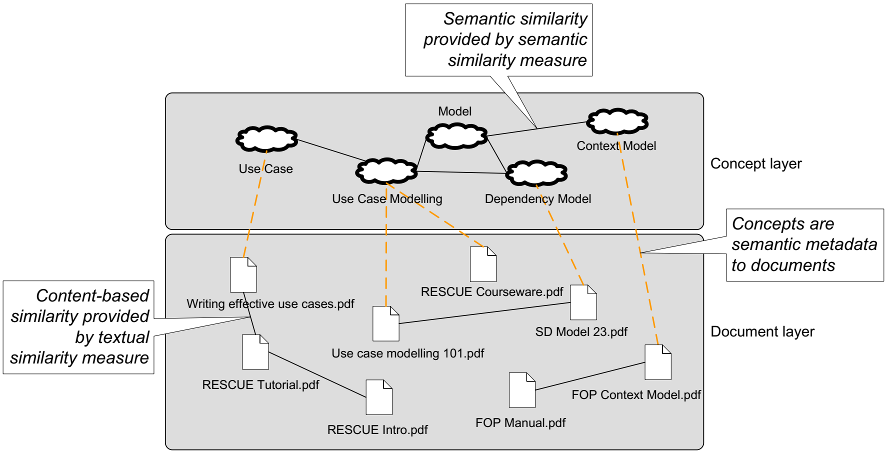

Fig. 1 The associative network consisting of of two interconnected layers

Finally, the network is searched using a spreading activation algorithm which combines spread of activation in the concept layer and spread of activation in the document layer (cf. section 3.5).

3.1 Calculating Semantic Similarity of Concepts

Concept nodes are associated in the concept layer by means of semantic similarity. For calculating the similarity of two ontological concepts a symmetric semantic similarity measure is used. The method was presented in [27] and requires two concepts belonging to the same ontology as input. It calculates the semantic similarity between these two concepts according to equation 1. This similarity measure builds on the path length to the root node from the least common subsumer (lcs) of the two concepts, which is the most specific concept they share as an ancestor. This value is scaled by the sum of the path lengths from the individual concepts to the root.

$$sim(c_1, c_2) = \frac{2 \cdot lcs(c_1, c_2)}{depth(c_1) + depth(c_2)} \tag{1}$$

With:
- c_1 ... first concept
- c_2 ... second concept
- lcs ... least common subsumer of two concepts
- $depth$ ... depth of concept in the class hierarchy

Depending on the features present in an ontology different similarity measures qualify to be applied. We chose the measure presented in [27], as a prominent feature of our ontology are taxonomic relations between concepts.

An advantage of the used measure is that it tries to address one of the typical problems of taxonomy-based approaches to similarity: relations in the taxonomy do not always represent a uniform (semantic) distance. The more specific the hierarchy becomes, the more similar a child node is to its father node in the taxonomy.

3.2 Calculating Text-Based Similarity of Documents

Document nodes are associated in the document layer by means of textual similarity. As similarity measure for text-documents we use an asymmetric measure based on the vector space model implemented in the open-source search-engine Lucene[2]. The similarity between two documents is calculated as shown in equation 2.

$$sim(d1, d2) = score(d1_{25}, d2) \tag{2}$$

With:

- $d1$ ... document vector of the first document
- $d2$ ... document vector of the second document
- $d1_{25}$ ... document vector of the first document with all term weights removed except the 25 highest terms weights

$d1_{25}$ is used as query vector for the *score*-measure of Lucene. For extracting the 25 terms with the highest weights, both the document content and the document title are taken into account. The calculation of Lucene's score is depicted in equation 3.

$$\begin{aligned} score(q, d) = {} & coord(q, d) \cdot queryNorm(q) \\ & \cdot \sum_{t\_in\_q} \left(tf(t\_in\_d) \cdot idf(t)^2 \cdot t.getBoost() \cdot norm(t, d) \right) \end{aligned} \tag{3}$$

With:

- q ... query vector
- d ... document vector
- $coord(q, d) = numberOfMatchingTerms/numberOfQueryTerms$
- $numberOfMatchingTerms$ ... number of terms in document matching query
- $numberOfQueryTerms$ ... number of terms in the query
- $queryNorm(q)$ ... normalization of the query vector, Lucene default used
- $tf(t\_in\_d)$ ... term frequency of current term in document, Lucene default used

[2] `http://lucene.apache.org/` (14.04.2008)

- $idf(t)$ ... inverse document frequency of current term in the document collection, Lucene default used
- $t.getBoost() = tf(t\_in\_q) \cdot idf(t)$
- $tf(t\_in\_q)$ ... term frequency of current term in query
- $norm(t, d) = 1/sqrt(numberOfDocumentTerms)$
- $numberOfDocumentTerms$ ... number of terms in the current document

Out of the various components that control the final score of a document matching a query, $coord(q, d)$ deserves special attention because it had shown in practice to contribute much to the final result. Thus a document that matches the set of query terms will be ranked higher than a document that only contains a smaller subset of all input query terms. Another important aspect of the scoring function is the document normalization factor, $norm(t, d)$. Documents that contain fewer terms will yield a higher score then long documents. This applies not only to the document content, but also to the document titles. Therefore the similarity of the title terms contributes more to the final score than the terms from the document body. On the other hand, the $t.getBoost()$ factor can be ignored in our case, because all query terms are weighted equally.

A detailed and a more in depth explanation of the various parameters that can be used to adapt the behavior of Lucene can be found in the Javadoc of the `org.apache.lucene.search.Similarity` class.

3.3 Semantic Annotation of Documents

The link between the two layers of the network is provided by annotations of resources with ontological concepts. As Handschuh [11] notes, different approaches to semantic annotation exist in literature. The author refers to [2] who differentiates between the following ways of semantic annotation:

- *Decoration*: Annotation of resources with a comment of the user.
- *Linking*: Annotation of resources with additional links.
- *Instance identification*: Annotation of resources with a concept. The annotated resource is an instance of the concept.
- *Instance reference*: Annotation of resources with a concept. The annotated resource references an individual in the world which is an instance of the concept.
- *Aboutness*: Annotation of resources with a concept. The annotated resource is about the concept.
- *Pertinence*: Annotation of resources with a concept. The annotated resource provides further information about the concept.

Semantic annotations in the present system are based on the *Aboutness* of resources. This means that we annotate whole documents with a set of concepts the content of the document is about. This is partly due to the usage of the current implementation inside the APOSDLE system. There annotations are used to express exactly the *aboutness* of resources and are formally described with the property *deals with* that is modeled inside the knowledge base of APOSDLE, that is used to store the semantic annotations (see section 4).

In approaches based on *Instance identification* or *Instance reference* as [6] or [14] annotation is treated on a more fine-grained level: Single words in documents are annotated with concepts stemming from the ontology.

We follow our approach for two reasons: (1) Although the complete semantics of words contained in a document are not recognized using this approach, the additional information added to the document still provides opportunities to be used at a later time in retrieving material [26], by a limited amount of human involvement. (2) We think that for the near future it makes sense to work on making the Semantic Web a reality, by focusing on bringing little semantics [13] into the current web and taking small steps. We follow this pragmatic approach and try to apply it to the Semantic Desktop in the context of our work.

3.4 Weighting the Annotations

In our (and other) approach(es) to semantic annotation, a document is either annotated with certain concepts or it is not. From a retrieval point of view this means that a document is either retrieved, if it is annotated with a concept present in the query, or it is not retrieved, if none of the concepts in the query are assigned to the document. Ranking the retrieved document set is impossible.

To allow for ranking the result set and to increase the performance of our service we weight the annotations between documents and concepts using a tf-idf-based weighting scheme. This is a standard instrument in information retrieval to improve retrieval results [19]. Our weighting approach is related to the one presented by [6], who are also weighting semantic annotations using a tf-idf-based measure.

$$weight(c,d) = tf(c,d) \cdot idf(c) = tf(c,d) \cdot \log \frac{D}{a(c)} \qquad (4)$$

With:

- c ... a concept
- d ... a document
- $tf(c,d)$ ... 1 if d is annotated with c, 0 otherwise
- $idf(c)$ ... inverse document frequency of concept c
- D ... total number of documents
- $a(c)$ ... number of documents annotated with concept c

3.5 Searching the Network

The network structure underlying the service is searched by spreading activation. Starting from a set of initially activated nodes in the network, activation spreads over the network and activates nodes associated with the initial set of nodes. Originally stemming from the field of cognitive psychology, where it serves as a model for operations in the human mind, spreading activation found its way over applications in both neural and semantic networks to information retrieval [8]. It is comparable to other retrieval techniques regarding its performance [16].

Beside systems that use spreading activation for finding similarities between text documents or search terms and text documents, approaches exist, which employ spreading activation for finding similar concepts in knowledge representations [1] [20]. The novelty of our approach lies in combining spreading activation search in a document collection with spreading activation search in a knowledge representation. The formula we use to calculate the spread of activation in our network is depicted in equation 5.

$$A(n_j) = \sum_{i=1}^{t} \frac{A(n_i) \cdot w_{i,j}}{\sum_{k=1}^{s} w_{i,k}} \tag{5}$$

With:

- $A(n_j)$ ... activation of node n_j
- $A(n_i)$ ... activation of node n_i
- t ... number of nodes adjacent to node n_j
- $w_{i,j}$ ... weight of edge between node n_i and node n_j
- s ... number of nodes adjacent to node n_i
- $w_{i,k}$ ... weight of edge between node n_i and node n_k

Search in our network is performed as follows:

1. Search starts with a set of concepts, representing the information need of the knowledge-worker. The concept nodes representing these concepts are activated.
2. *Optionally*, activation spreads from the set of initially activated concepts over the edges created by semantic similarity to other concepts nodes in the network.
3. Activation spreads from the currently activated set of concept nodes to the document nodes over the edges created by semantic annotation to find documents that deal with the concepts representing the information need.
4. *Optionally*, activation spreads from the documents nodes currently activated to document nodes that are related by means of textual similarity and are therefore associated with the document nodes.
5. Those documents corresponding to the finally activated set of document nodes are returned as search result to the user.

4 Implementation Inside the APOSDLE Project

The associative network structure and the spreading activation algorithm presented in section 3 have been implemented to support the retrieval of resources inside the first prototype of the APOSDLE system.

The goal of the current version of the APOSDLE system is to help knowledge-workers understanding the field of requirements engineering. In order to meet its goals APOSDLE uses a knowledge base in the form of a domain ontology, which described the field of requirements engineering in which the first prototype of APOSDLE operates, and a document base, which contains learning material (definitions, examples, tutorials, etc.) about requirements engineering that are partly annotated with concepts from the domain ontology.

The domain ontology consists of 70 concepts, 21 of which are used to annotate documents. The document base consists of 1016 documents, 496 documents of which are annotated with one or more concepts from the knowledge base. As we can see the scenario of APOSDLE provides a typical example of scarce annotations: only parts of the ontology are used for annotation and only parts of the documents are annotated. We see this setting corresponding to the coverage problematic presented in section 1 and employing associative retrieval techniques appropriate to finding relevant material that was not originally annotated with concepts from the domain ontology.

The service implemented in the APOSDLE project and presented in this section relies on knowledge contained in an ontology and the statistical information in a collection of documents. The service is queried with a set of concepts from the ontology and returns a set of documents. Documents in the system are (partly) annotated with ontological concepts if a document *deals with* a concept. For example, if the document is an introduction to use case models it is annotated with the corresponding concept in the ontology. In APOSDLE, the annotation process is performed manually but is supported by statistical techniques (e.g. identification of frequent words in the document collection) [18].

Concepts from the ontology are used as metadata for documents in the system. Opposed to classical metadata, the ontology specifies relations between the concepts. For example, class-subclass relationships are defined as well as arbitrary semantic relations between concepts are modeled (e. g. `UseCase isComposedOf Action`). The structure of the ontology has been used for calculating the similarity between two concepts in the ontology according to the measure presented in section 3.1. This similarity has been used to expand a query with similar concepts before retrieving documents dealing with a set of concepts. After retrieval of documents was performed, the result set was expanded by means of textual similarity as introduced in section 3.2. The implementation of a specific associative network inside the APOSDLE system has allowed developing and testing different combinations of query and result expansion that are based on the spreading activation algorithm presented in section 3.5. The next section contains an evaluation of the performance of different combinations and a discussion of the results obtained.

5 Evaluation

In this section we describe the evaluation that we performed. We talk about the evaluation measures, the queries used for evaluation, how we collected relevance judgments and about the service configuration rankings obtained.

5.1 Semantic Web Information Retrieval and Evaluation

At present information retrieval in the Semantic Web (on the Semantic Desktop) is an inhomogeneous field (c.f. [25]. Although a good amount of approaches does exist, different information is used for the retrieval process, different input is accepted and different output is produced. This complicates to define generally applicable rules for the evaluation of an information retrieval system for the Semantic Web (or the Semantic Desktop) and to create a test collection for this application area of information retrieval.

The present approach to retrieval on the Semantic Desktop is different from current attempts to retrieval in a desktop environment: (1) the semantic information present in an ontology is taken into account for retrieval purpose; (2) the query to the retrieval service is formulated by a set of concepts stemming from an ontology as opposed to a set of terms (words) as typically used in the context of desktop search. As we are not aware of any standard test corpora for the evaluation of an information retrieval service for the Semantic Desktop we have created our own evaluation environment.

5.2 The Test Corpus

A major obstacle in the easy evaluation of Semantic Web technology based information retrieval systems is the absence of standardized test corpora, as they exist for text-based information retrieval.

Therefore we have built our own test corpus based on the data available in the first release of the APOSDLE system [15]. The first version of APOSDLE was built for the domain of Requirements Engineering. This resulted into a domain ontology for this field and a set of documents dealing with various topics of Requirements Engineering. The document base was provided by a partner in the APOSDLE project, with expertise in the field of Requirements Engineering, while the ontology was modeled by another partner. Together these two partners sign responsible for the annotation of the document base with concepts from the ontology. The ontology contains 70 concepts and the document set consists of 1016 documents. 496 documents were annotated using one or more concepts. 21 concepts from the domain ontology were used to annotate documents.

In its size our test collection is comparable to test collections from early information retrieval experiments as the Cranfield or the CACM collections[3].

In addition to the absence of corpora for Semantic Web information retrieval we are unaware of any standard text-retrieval corpora for evaluating a service with characteristics similar to ours. We considered treating the ontological concepts used for querying our service equivalent to query terms of a text-retrieval system to be able to use a standard corpus. Therefore we would have needed some structure relating the terms contained in the documents, as it is the case with the ontology in our system which relates concepts. For this task we could have used a standard thesaurus. As this knowledge structure is different to the ontology originally used (and therefore different similarity measures had to be applied to it), this would have led us to evaluating a service with different properties than our original one.

We also considered the INEX[4] test collection for evaluating our service. INEX provides a document collection of XML documents which would have provided us with textual data associated with XML structure information. Unfortunately again an ontology relating the metadata used as XML markup is unavailable. This would have prevented us from employing (and evaluating) the functionality provided by the query expansion technique, which founds on the ontology.

5.3 Measures Used for Evaluation

The central problem in using classic IR measures as *recall* or *mean average precision* is that they require complete relevance judgements, which means that every document is judged against every query [4]. [10] notices that recall can not be determined precisely with reasonable effort. Finally [5] states that: *Building sets large enough for evaluation of realworld implementations is at best inefficient, at worst infeasible.*

Therefore we opted for using evaluation measures that do not require hat every document is judged against every query. We decided for using precision (P) at rank 10, 20 and 30. In addition we made use of infAP [28] which approximates the value of average precision (AP) using random sampling.

For calculating the evaluation scores we have used the `trec_eval`[5] package, which origins from the Text REtrieval Conference (TREC) and allows for calculating a large number of standard measures for information retrieval system evaluation.

[3] `http://www.dcs.gla.ac.uk/idom/ir_resources/test_collections/` (14.04.2008)
[4] `http://inex.is.informatik.uni-duisburg.de/` (14.04.2008)
[5] `http://trec.nist.gov/trec_eval/` (14.04.2008)

5.4 Queries Used for Evaluation

The queries that were used for the evaluation of the service are formed by sets of concepts.

The first version of the APOSDLE system presents resources to knowledge workers to allow them to acquire a certain competency. To realize search for resources that are appropriate to build up a certain competency, competencies are represented by sets of concepts from the domain ontology. These sets are used as queries for the search for resources. For the evaluation of the APOSDLE system all distinct sets of concepts representing competencies[6] were used as queries. In addition all concepts from the domain model not already present in the set of queries were used for evaluation purposes.

5.5 Collecting Relevance Judgments

8 different service configurations were tested and compared against each other based on the chosen evaluation measures. 79 distinct queries were used to query every service configuration. Queries were formed by sets of concepts stemming from the domain ontology.

For every query and service configuration the first 30 results were stored in a database table, with one row for every query-document pair. Query-document pairs returned by more than one service configuration were stored only once. The query-document pairs stored in the database-table were then judged manually by a human assessor. All query-document pairs were judged by the same person. The assessor was not involved in defining the competency to concept mappings uses as queries (c.f. section 5.4).

After relevance judgment, both, the results obtained by the different service configurations and the global relevance judgments have been stored into text files in a format appropriate for the `trec_eval` program. We then calculated the P(10), P(20), (P30) and infAP scores for the different service configurations.

5.6 The Obtained Service Configuration Ranking

Table 1 shows the calculated P(10), P(20), (P30) and infAP scores for the different service configurations. The columns *SemSim*, *TxtSim* indicate whether semantic similarity or text-based similarity was used for the search. Table 2 shows the service configuration ranking based on the obtained evaluation scores.

Configuration 1 (conf_1) is the baseline configuration of our service. The results delivered by this configuration are comparable to the use of a query language as SPARQL combined with an idf-based ranking (based on documents annotated with concepts) and no associative retrieval techniques used.

[6] Different competencies can be represented by the same concepts.

Table 1 Evaluation scores of service configurations calculated using P(10), P(20), P(30) and infAP

Conf.	SemSim	TxtSim	P(10)	P(20)	P(30)	infAP
conf_1	No	No	0.2418	0.2051	0.1700	0.1484
conf_2	No	Yes	0.3089	0.2778	0.2502	0.2487
conf_3	Yes (> 0.5)	No	0.3165	0.2608	0.2131	0.2114
conf_4	Yes (> 0.7)	No	0.3114	0.2582	0.2097	0.2001
conf_5	Yes (> 0.5)	Yes	0.3848	0.3405	0.3046	0.3253
conf_6	Yes (> 0.7)	Yes	0.3924	0.3494	0.3089	0.3326

Table 2 Ranking of service configurations based on P(10), P(20), P(30) and infAP

Rank	P(10)	P(20)	P(30)	infAP
1 (best)	conf_6	conf_6	conf_6	conf_6
2	conf_5	conf_5	conf_5	conf_5
3	conf_3	conf_2	conf_2	conf_2
4	conf_4	conf_3	conf_3	conf_3
5	conf_2	conf_4	conf_4	conf_4
6 (worst)	conf_1	conf_1	conf_1	conf_1

Exactly those documents are retrieved that are annotated with the concepts present in the query.

All other configurations make use of query expansion based on semantic similarity or result expansion based on text-based similarly. Configurations 3, 4, 5 and 6 perform query expansion. Configurations 2, 5 and 6 perform result expansion.

All associative search approaches employing semantic similarity (configurations 3, 4, 5 and 6), text-based similarity (configurations 2, 5 and 6) or both (configurations 2, 3, 4, 5 and 6) increase retrieval performance compared to the baseline (configuration 1). Additional relevant documents are found, which are not annotated with the concepts used to query the service.

5.7 Discussion

We now discuss the evaluation measures used and why we think that the amount of relevance judgments collected is sufficient for a proper evaluation of our service.

5.7.1 P(10), P(20) and P(30)

[3] evaluate the stability of evaluation measures. They calculate the error rate of measures based on the number of errors occurring whilst comparing two systems using a certain measure. They divide the number of errors by the total number of possible comparisons between two different systems.

Based on previous research they state that an error rate of 2.9% is minimally acceptable. They find that P(30) exactly reaches this error rate of 2.9% in their experiment with 50 queries used. Finally they suggest that the amount of queries should be increased for P(n) measures, where $n < 30$. And suggest that 100 queries would be safe if the measure P(20) is used.

We performed our experiment with 79 distinct queries and used the measures P(10), P(20) and P(30). Following the results of [3] the size of our query set should be appropriate for P(30). We are fortified in this assumption as the ranking of the 8 service configurations is identical for P(20), P(30) and infAP.

5.7.2 infAP

The Trec 8 Ad-Hoc collection consists of 528,155 documents and 50 queries which make a total amount of 26,407,750 possible relevance judgments. 86830 query-document relevance pairs are actually judged. This set of pairs is created by depth-100 pooling of 129 runs. Therefore 0.33% of the possible relevance judgments are performed.

Our collection consists of 1026 documents and 79 queries, which results in a total of 81,054 possible relevance judgments. This set of pairs is created by depth-30 pooling of 8 runs and 498 additional relevance judgments that were performed for runs that were not part of the experiment. 1938 query document pairs were actually judged. Therefore 2.39% of all possible relevance judgments were performed.

The depth-100 pool for the 8 evaluated runs would consist of 4138 query-document pairs. As we judged 1938 query-document pairs, we judged 46.83% of our potential depth-100 pool. [28] report a Kendall's tau based rank correlation of above 0.9 between infAP and AP with as little as 25% of the maximum possible relevance judgments of the depth-100 pool of the Trec 8 Ad-Hoc collection. They consider two rankings with a rank correlation of above 0.9 as equivalent.

With 46.83% of our potential depth-100 pool judged, we are confident that the infAP measure produces an estimation sufficiently accurate. Again our confidence in the results of infAP is assured by the equivalence of the ranking of the 8 service configurations for P(20), P(30) and infAP.

6 Related Work

Beagle++ [7] is a search engine for the Semantic Desktop and indexes RDF-metadata together with document content. Both [6] and [14] present an extension of the vector space model. Together with document content they index semantic annotations of documents and use this information for search. All three are very promising approaches that extend the vector space model using semantic information. None of them employs measures of semantic association.

[20] present a hybrid approach for searching the (semantic) web, they combine keyword based search and spreading activation search in an ontology for search on websites. Ontocopi [1] identifies communities of practice in an ontology using spreading activation based clustering. Both are prospective approaches employing ontology-based measures of association and evaluating them using spreading activation. They do not integrate text-based measures of association into their systems.

7 Conclusions and Future Work

We have presented an information retrieval service for the Semantic Desktop, which is based on techniques from associative information retrieval. We have evaluated the presented service using standard measures for information retrieval system evaluation. As classic measures for evaluation as recall and average precision require that every document is judged for every query we have chosen precision at ranks 10, 20 and 30 as evaluation measures. In addition we made use of the random sampling approach performed by the infAP measure. Following recent works [4] [28] in information retrieval system evaluation we are confident that our chosen approach reflects the actual relation between the service configurations as the ranking of the service configurations remains identical for the measures P(20), P(30) and infAP.

Our experiments encourage us, that the application of associative retrieval techniques to information retrieval on the Semantic Desktop is an adequate strategy. We tend to conclude that text-based methods for associative retrieval result in a higher increase in retrieval performance, therefore we want to explore the approach of attaching a set of terms to every concept in our domain ontology during modeling time to provide search results even for concepts that are not used for annotation. In addition we want to extend our research towards the application of different semantic similarity measures within our service.

Acknowledgments

While a shorter version of this paper focusing on the description of the retrieval approach was presented at I-SEMANTICS 2007 an in depth description of the approach to the evaluation of the developed service was presented at FGIR 2007 / LWA 2007. We thank the anonymous reviewers of our submissions at I-SEMANTICS 2007 and LWA 2007 for their constructive feedback.

This work has been partially funded under grant 027023 in the IST work programme of the European Community. The Know-Center is funded by the Austrian Competence Center program Kplus under the auspices of the Austrian Ministry of Transport, Innovation and Technology (`www.ffg.at/index.php?cid=95`) and by the State of Styria.

References

1. Alani, H., Dasmahapatra, S., O'Hara, K., Shadbolt, N.: Identifying communities of practice through ontology network analysis. IEEE Intell. Syst. 18(2), 18–25 (2003)
2. Bechhofer, S., Goble, C.: Towards annotation using DAML+OIL. In: K-CAP 2001 workshop on Knowledge Markup and Semantic Annotationq (2001)
3. Buckley, C., Voorhees, E.M.: Evaluating evaluation measure stability. In: SIGIR 2000: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 33–40. ACM Press, New York (2000)
4. Buckley, C., Voorhees, E.M.: Retrieval evaluation with incomplete information. In: SIGIR 2004: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 25–32. ACM Press, New York (2004)
5. Carterette, B., Allan, J., Sitaraman, R.: Minimal test collections for retrieval evaluation. In: SIGIR 2006: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 268–275. ACM Press, New York (2006)
6. Castells, P., Fernández, M., Vallet, D.: An adaptation of the vector-space model for ontology-based information retrieval. IEEE Trans. Knowl. Data Eng. 19, 261–272 (2007)
7. Chirita, P.-A., Costache, S., Nejdl, W., Paiu, R.: Beagle++: Semantically enhanced searching and ranking on the desktop. The Semantic Web: Research and Applications (2006)
8. Crestani, F.: Application of spreading activation techniques in information retrieval. Artif. Intell. Rev. 11, 453–482 (1997)
9. Decker, S., Frank, M.R.: The networked semantic desktop. In: WWW Workshop on Application Design, Development and Implementation Issues in the Semantic Web (2004)
10. Fuhr, N.: Information Retrieval: Skriptum zur Vorlesung im SS 06 (December 19, 2006)
11. Handschuh, S.: Semantic Web Services: Concepts, Technologies, and Applications. In: Semantic Annotation of Resources in the Semantic Web, pp. 135–155. Springer New York, Inc., Secaucus (2007)
12. Heflin, J., Hendler, J.: Searching the web with Shoe. In: Artificial Intelligence for Web Search. Papers from the AAAI Workshop. WS-00-01 (2000)
13. Hendler, J.: The dark side of the semantic web. IEEE Intell. Syst. 22, 2–4 (2007)
14. Kiryakov, A., Popov, B., Terziev, I., Manov, D., Ognyanoff, D.: Semantic annotation, indexing, and retrieval. Journal of Web Semantics 2, 49–79 (2004)
15. Lindstaedt, S.N., Mayer, H.: A storyboard of the aposdle vision. In: Nejdl, W., Tochtermann, K. (eds.) EC-TEL 2006. LNCS, vol. 4227, pp. 628–633. Springer, Heidelberg (2006)
16. Mandl, T.: Tolerantes Information Retrieval. Neuronale Netze zur Erhöhung der Adaptivität und Flexibilität bei der Informationssuche. PhD thesis, University of Hildesheim (2001)
17. McCool, R.: Rethinking the semantic web, part 1. IEEE Internet Comput. 9(6), 88–97 (2005)

18. Pammer, V., Scheir, P., Lindstaedt, S.N.: Two protégé plug-ins for supporting document-based ontology engineering and ontological annotation at document level. In: 10th International Protégé Conference, Budapest, Hungary (July 15-18, 2007)
19. Robertson, S.E., Spärck Jones, K.: Simple, proven approaches to text retrieval. Technical report, University of Cambridge, Computer Laboratory (1994)
20. Rocha, C., Schwabe, D., de Aragão, M.P.: A hybrid approach for searching in the semantic web. In: Proceedings of the 13th international conference on World Wide Web, WWW 2004 (2004)
21. Sabou, M., d'Aquin, M., Motta, E.: Using the semantic web as background knowledge for ontology mapping. In: International Workshop on Ontology Matching, OM-2006 (2006)
22. Salton, G.: Associative document retrieval techniques using bibliographic information. JACM 10, 440–457 (1963)
23. Salton, G.: Automatic Information Organization and Retrieval. McGraw Hill, New York (1968)
24. Sauermann, L., Bernardi, A., Dengel, A.: Overview and outlook on the semantic desktop. In: Proceedings of the 1st Workshop on The Semantic Desktop at the ISWC 2005 Conference (2005)
25. Scheir, P., Pammer, V., Lindstaedt, S.N.: Information retrieval on the semantic web - does it exist? In: LWA 2007, Lernen - Wissensentdeckung - Adaptivität, September 24–26, 2007. Halle/Saale (2007)
26. Spärck Jones, K.: What's new about the semantic web?: some questions. SIGIR Forum 38, 18–23 (2004)
27. Wu, Z., Palmer, M.S.: Verb semantics and lexical selection. In: Meeting of the Association for Computational Linguistics (ACL), pp. 133–138 (1994)
28. Yilmaz, E., Aslam, J.A.: Estimating average precision with incomplete and imperfect judgments. In: CIKM 2006: Proceedings of the 15th ACM international conference on Information and knowledge management, pp. 102–111. ACM Press, New York (2006)