Semantic Service Retrieval based on Natural Language Querying and Semantic Similarity

Richard Eckart de Castilho Iryna Gurevych
Ubiquitous Knowledge Processing (UKP) Lab
Technische Universität Darmstadt
Darmstadt, Germany
http://www.ukp.tu-darmstadt.de

***Abstract*—In this paper, we address the task of semantic service retrieval based on natural language queries. We analyze identifiers of services, operations, and parameters extracted from WSDL service descriptions with respect to their semantic content. In order to measure the semantic similarity between query and service description, we introduce a novel computationally efficient document similarity measure based on information content and fuzzy set theory.**

I. Introduction

The internet is probably the largest marketplace there is. One growing segment of that marketplace is the trade with web services. Large service repositories help bringing service customers and service providers together. Currently one of the largest web service repositories, SeekDa,[1] claims to index over 28,000 services. These services are of a wide variety from validating email over looking up literature to booking flights.

In order to retrieve services from such a web service repository, a potential *service consumer* poses a *service request*. The repository search engine employs a service retrieval method to find services fitting the functional needs expressed in the service request and ranks them by relevance.

The standard format for web service descriptions is the *Web Service Description Language*[2] (WSDL). However, an end consumer will not interact with a web service on the low technical level WSDL was designed for. It is more likely for an end consumer to interact with a front-end application, e.g. a web market place. Thus, instead of looking for *a service to look up the price of a book*, the consumer may request *find the price of Lord of the Rings as hardcover*. The web market place then should retrieve services capable of finding the price of the book and invoke these to retrieve the actual price to be displayed to the consumer.

A. Service Descriptor

We take a brief look at which kind of information we find in WSDL service descriptions (see figure 1).

Identifiers are names used to identify the service itself, operations offered by the service as well as data types and parameters passed to and returned from these methods. These identifiers carry a significant amount of information about the semantics of the entity they identify [1]. Like in programming languages, identifiers usually have to appear as a single token and, thus, there are a number of conventions how to deal with multi-word identifiers. A prominent example is *CamelCase* which concatenates multiple words into a single token while capitalizing the first letter of each word. A service retrieval engine should be aware of such conventions, e.g. in order to correctly tokenize *getBookPrice* as *get*, *book*, *price*.

Data types can be primitive, e.g. integer, string or boolean, or can be structured, consisting of a number of fields, each carrying a name and type.

Signatures define which are the input and output parameters of operations. Each parameter carries a name and a type.

Documentation may be included in service descriptions, however, most of the time it is not [2]. When present, it is often very rudimentary, for example a *getBookPrice* operation may simply be described as *Gets the price of a book*.

```
Service: AuthorBookmaxpriceService
  Operation: get_BOOK_MAXPRICE
    Input Parameter: get_BOOK_MAXPRICERequest
      _AUTHOR   (AuthorType)
    Output Parameter: get_BOOK_MAXPRICEResponse
      _MAXPRICE (MaxPriceType)
      _BOOK     (BookType)
      isTitled  (Title)
      hasType   (Book-Type)
      writtenBy (AuthorType)
```

Fig. 1. Information from a WSDL service description

B. Task Description

A major problem in service retrieval is the *vocabulary gap*. The problem occurs when service request and service offer use different vocabularies. Bag-of-words based retrieval models lead to unsatisfactory results in this case, because they require request and offer to share the same terms (section II-A). This problem occurs due to various causes, e.g. due to a difference in the level of abstraction (*Lord of the Rings* vs. *book*) or due to the use of synonyms (*hardcover* vs. *hardback*). We expect the vocabulary gap to occur particularly often in the kind of scenario mentioned earlier: a consumer states a concrete request to a front-end application; the front-end tries to locate services that can answer to the request; finally the front-end invokes the service on behalf of the consumer.

Semantic retrieval models address the vocabulary gap by utilizing semantic resources which link terms to concepts.

[1] http://webservices.seekda.com/

[2] Web Service Description Language, http://www.w3.org/TR/wsdl

Thus, it becomes possible to calculate the similarity at the level of concepts and find a service offer matching the request if they contain no shared terms but are semantically related.

Additionally, service retrieval has particular demands towards the retrieval model. Information retrieval models often consider documents containing information beyond what was requested in the query as less relevant than documents containing only information stated in the query. In service retrieval, though, it should not have a negative impact if a service provides functionality beyond what was requested. We will later refer to this as the *particular nature* of service retrieval.

Looking at several existing semantic retrieval models (section II-B), we notice they belong to either of two classes:

1) *corpus-based retrieval models* (section II-B1) use computationally efficient vector space models that derive semantic similarity from term co-occurrences but allow little control over the semantic relations deduced;
2) *knowledge-based retrieval models* (section II-B2) use path-based similarity measures that work with an explicit semantic resource allowing detailed control over the semantic relations such as hypernymy, hyponymy, and synonymy, but tend to be computationally inefficient.

Our contribution is a computationally efficient knowledge-based semantic retrieval model using fuzzy sets (section III), which we experimentally evaluate in the context of service retrieval (sections IV and V). Being knowledge-based, our model allows full control over the semantic relations used in the retrieval process. This can be helpful e.g. to fine-tune the retrieval to a particular domain such as literature or medicine or even to the product catalog of a vendor.

Most work on service retrieval, like [3]–[5], assumes that a service request is posed using the same formalism that is also used for the service description – in our scenario that would be WSDL. Thus, these approaches can exploit the structured nature of the service description and e.g. search for optimal matches between operations specified in the service request and operations specified in the service offer. Compared to this, the information present in a natural language query is quite limited. This should be taken into account when comparing our work with the state-of-the-art in service retrieval.

II. Service Retrieval Models

At the core of service retrieval, as a kind of information retrieval, is the *retrieval model* which is used to match documents to a query and to rank them by relevance. In the context of this paper, the documents to be matched and ranked are service offers in the form of WSDL service descriptions while the queries are service requests in the form of natural language.

A. Term-based Information Retrieval Models

This section outlines two popular term-based information retrieval models – the Standard Boolean Model and the Vector Space Model – which represent a strong baseline and allow for short query response times. Both models, however, have the drawback that a document can only be retrieved if the query contains at least one of the document's terms. Thus there is a *vocabulary gap* if a document should semantically match the query, but does not because it does not contain a query term.

The *Standard Boolean Model* (SBM) of information retrieval is based on set-theory. Both, query and document are interpreted as sets of terms. A document matches the query if the intersection of their respective sets of terms is non-empty. The model allows complex boolean queries using the boolean operators *and*, *or* and *not*. This model does not allow the query writer to weight the query terms, nor does it take into account that some documents matching the query may be more relevant than others.

The *Vector Space Model* (VSM, [6]) is a retrieval model that treats queries and documents as sets of terms represented as vectors in an n-dimensional space. The elements of these vectors represent the relevance or importance of a term. The relevance of a term to a document $r_t(d)$ is commonly calculated by multiplying the frequency of the term in the document TF with the inverse document frequency IDF, which is the logarithm of the number of documents containing the term [7].

$$r_t(d) \quad = \quad TF(d,t) * IDF(t)$$

The importance of a query term is typically 1 unless explicitly stated otherwise by the query writer. The similarity between query and document is calculated by a vector product, typically the cosine.

Using a VSM to retrieve services based on identifiers and documentation contained in the service descriptions has been recognized as providing a good baseline for service retrieval and is incorporated by various state-of-the-art approaches, e.g. [2]–[4], [8].

The SBM and the VSM are often combined. When adding a document to an information retrieval system based on a combined approach, documents are indexed twice. The first index is a reverse document index that allows to quickly look up those documents containing at least one query term, which is used to search documents using the Standard Boolean Model. The second index stores the term vectors for each document. This index is used to rank documents according to the VSM. Together, the indexes allow for short query response times.

B. Semantic Service Retrieval Models

While the traditional VSM is based on sets of terms, there are also alternative models that employ vectors representing sets of concepts – we will refer to these as *semantic vector space models* (SVSM). Semantic retrieval models address the vocabulary gap by allowing a document to be retrieved if it shares a common concept with the query. For example, a query term *hardcover* would match a document term *hardback* as both terms are synonymous, representing the same concept.

Corpus-based methods use the semantic relatedness implicitly induced by the fact that semantically related terms often occur together. By automatically analyzing term co-occurrences in a sufficiently large document collection, they

aim to provide an objective semantic similarity measure. *Knowledge-based* methods on the other hand, use detailed explicit semantic relations modeled by a knowledge engineer and encoded in a semantic resource such as an ontology.

1) Corpus-based Retrieval Models: Latent Semantic Analysis (LSA, [9]) is a corpus-based SVSM which uses singular value decomposition to reduce the dimensionality of traditional VSM term vectors. The process conflates dimensions associated with terms commonly co-occurring in a training corpus into a single dimension which is thought to represent the concept of which the terms are representatives. For example, it is likely that LSA folds the synonymous terms *student* and *pupil* into the same dimension. Thus, the process converts term vectors to concept vectors. The concept vector for a document is calculated by summing up the concept vectors associated with each of the document terms. The dimensionality of the LSA vectors needs to be defined prior to indexing. Landauer et al. [9] found 300 dimensions to yield the best results on a document collection with 60.000 unique words.

Explicit Semantic Analysis (ESA, [10]) is another corpus-based SVSM which assumes that terms co-occurring in documents frequently are semantically related. Further, it assumes that each corpus document focusses on a particular idea and thus can be interpreted as a concept. Consequently, the dimensionality of an ESA vector equals the number of documents in the corpus. Generating a concept vector for a set of terms is a two-step process in ESA: 1) a term concept vector is formed in which the elements are TF*IDF values of the terms in each document; 2) a document concept vector is computed as the centroid of the term concept vectors for each term occurring in the document. The fact that the dimensionality of the ESA vector space corresponds to the number of documents in the corpus can easily become an issue for large corpora.

2) Knowledge-based Retrieval Models: Exploiting an explicit semantic resource such as a thesaurus or an ontology typically means that the resource is used to calculate the semantic relatedness between two concepts, one represented by a query term, and the other by a document term. Applicable term relatedness measures have been introduced by [11]–[15]. Each of these measures calculates the relatedness in terms of a path of hypernym/hyponym relations connecting two concepts.

To compare two sets of terms using these measures, an aggregation strategy is required, such as averaging scores or weighted bipartite graph matching (WBPGM). WBPGM is the problem of finding a pairwise matching between the elements of two sets A and B such that the sum of a function $w : A \times B \rightarrow \mathbb{R}$ is optimized and that no element of either set is paired more than once. Knowledge-based approaches in combination with algorithms for WBPGM have been used in particular for signature matching where it is expected that each input/output parameter in the service request should have exactly one corresponding input/output parameter in the service offer [4], [5].

Models using aggregated pair-wise term scores do not share the characteristics that make VSM or SVSM computationally efficient. Because the aggregated relatedness score is calculated from the pair-wise relatedness scores of each combination of query and document terms, it is not possible to pre-calculate any information on a per-document level, as is the case for the VSM and SVSMs. Instead, all pair-wise relatedness scores within the knowledge-base need to be pre-calculated. Unfortunately, the number of required computations grows quadratically with the size of the knowledge base and quickly becomes unmanageable.

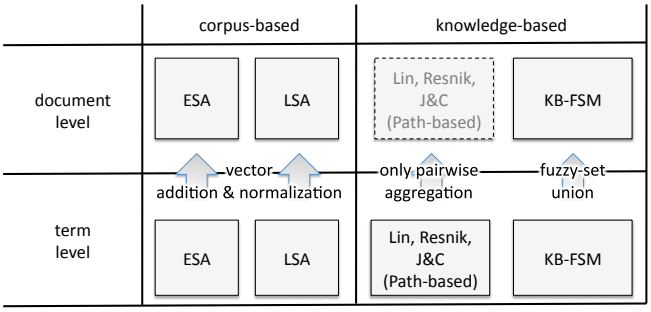

Fig. 2. Strategies to generalize term measures to document measures (dashed line indicates models that were not intended to be used at the document level)

Still, it is desirable to be able to efficiently employ knowledge-based approaches as they offer a great degree of control over the semantic matching process. As previously stated, they allow, for example, to fine tune the matching process to a particular domain such as literature or medicine or even to the product catalog of a vendor.

III. Knowledge-based Fuzzy Set Model

We propose a computationally efficient knowledge-based semantic retrieval model based on fuzzy sets that combines the computational efficiency of a VSM with the control over semantic relations offered by a knowledge-based approach – we call it *Knowledge-based Fuzzy Set Model* (KB-FSM). In contrast to other knowledge-based models (cf. section II-B2), which first calculate pair-wise similarities between query and document terms and then aggregate over the scores, our approach produces an aggregate fuzzy-set representation for each document (section III-B), without knowing the query. This provides computational efficiency similar to that of VSMs because the fuzzy concept sets can be pre-calculated for each document. We use Dice's coefficient (section III-C) to calculate the similarity between the fuzzy sets derived from queries and documents and later also experiment with a slight variation of that coefficient (section V-D). A schematic comparison of the aggregation strategies used by our approach and by the other retrieval models mentioned in section II is given in figure 2.

A. Semantic Representation on the Term-Level

Like the knowledge-based term relatedness measures by [13]–[15], our method is based on the idea of information content (IC). IC measures a concept's specificity, e.g. the very general concept *thing* has a very low information content while that of the more specific concept *book* is much higher. One way of measuring the IC is using a knowledge base K. The

intrinsic information content IIC [16] determines the IC of a concept c based on the number of its hyponyms $h(c)$ compared to the total number of concepts in the knowledge base K. The intuition here is that a term that has many hyponyms is less specific (contains less information) than a term that has few hyponyms or none at all. It is defined as a function $IIC : C \rightarrow [0, 1]$:

$$IIC(c) = 1 - \frac{log(|h(c)| + 1)}{log|K|}$$

We turn to creating a semantic representation of a single term using IC. Assume a query Q and a set of documents D_1, D_2, and D_3 each containing a single term:

$$\begin{aligned} Q &: \textit{'book'} \\ D_1 &: \textit{'book'} \\ D_2 &: \textit{'fantasy'} \\ D_3 &: \textit{'hardcover'} \end{aligned}$$

In addition, assume a knowledge base defining the concepts *book* and *fantasy* as subclasses of *thing* and *hardcover* as a subclass of *book*. The IC is noted in parentheses.

```
                ,- fantasy (1.0)
thing (0.0) {
                `- book (0.5) -- hardcover (1.0)
```

Obviously, document D_1 is more relevant to the query Q than the document D_2. We also take D_1 to be more relevant to the query than D_3 since we assume that there was a good reason for the query to refer to the more general concept *book* and not to *hardcover*. To reflect this, template-based approaches (e.g. [2]) build separate vectors for e.g. direct hypernyms, direct hyponyms and siblings of concepts and assign different weights to each of these vectors. Our method represents this idea using the *link strength* (LS) introduced by [15]. The link strength $LS(c, h)$ is calculated as the difference of the IC of a concept c and some hypernym/hyponym h.

$$LS(c, h) \quad = \quad |IC(c) - IC(h)|$$

As it is defined, the LS is a *distance measure*, but we are interested in a *similarity measure* to determine how close a term and its hypernym are in terms of IC. Therefore, we define the *inverse link strength* (ILS) as:

$$ILS(c, h) \quad = \quad 1 - LS(c, h)$$

To get a semantic representation of a term, we now calculate the ILS between the term and all of its hypernyms. For the purpose of building this representation, we assume the hypernym relation to be reflexive and also include the concept representing the term itself. For the document D_3 containing the term *hardcover* this yields the following results:

$$\begin{aligned} 0.0 &= ILS(\text{'hardcover'}, \text{'thing'}) \\ 0.5 &= ILS(\text{'hardcover'}, \text{'book'}) \\ 1.0 &= ILS(\text{'hardcover'}, \text{'hardcover'}) \\ D_3 &\rightarrow (\text{'thing'}(0.0), \text{'book'}(0.5), \text{'hardcover'}(1.0)) \end{aligned}$$

This representation, that we call the *semantic context* of the term, constitutes a *fuzzy set* [17]. In a fuzzy set, each set element $x \in A$ has a degree of membership in the set A written as $A(x)$. In the semantic context of a term, the ILS score calculated between the term and one of its hypernyms represents the degree of membership of that particular hypernym. The semantic contexts for the query and each of the documents are consequently

$$\begin{aligned} Q &\rightarrow (\text{'thing'}(0.0), \text{'book'}(1.0)) \\ D_1 &\rightarrow (\text{'thing'}(0.0), \text{'book'}(1.0)) \\ D_2 &\rightarrow (\text{'thing'}(0.0), \text{'fantasy'}(1.0)) \\ D_3 &\rightarrow (\text{'thing'}(0.0), \text{'book'}(0.5), \text{'hardcover'}(1.0)) \end{aligned}$$

B. Aggregation to Document-Level

We now extend this approach to documents containing more than one term by examining how to aggregate the semantic contexts of two terms. If the two terms share a common hypernym, both induce an ILS score for it. For the aggregated semantic context, we choose the highest of these ILS scores for each concept in the original semantic contexts.

This corresponds to the fuzzy union operation $(A \cup B)(x)$, which directly reflects our intention of choosing the highest ILS score. To illustrate this, we introduce a document D_4 : 'fantasy book' being the concatenation of D_2 and D_1:

$$\begin{aligned} D_4 &: \text{'fantasy book'} \\ D_4 &\rightarrow (D_2 \cup D_1)(x) \\ &\rightarrow (\text{'thing'}(0.0), \text{'fantasy'}(1.0), \text{'book'}(1.0)) \end{aligned}$$

C. Calculation of Semantic Similarity

To calculate the similarity between two concept vectors, we again apply the fuzzy set theory. The Dice coefficient calculates the similarity of two sets using the set union, set intersection, and set cardinality operations – all of which are defined on fuzzy sets:

$$\begin{aligned} (A \cap B)(x) &= min[A(x), B(x)] \\ (A \cup B)(x) &= max[A(x), B(x)] \\ |A|(x) &= \sum A(x) \end{aligned}$$

The Dice coefficient on fuzzy sets d_f therefore is

$$\begin{aligned} d_f(A(x), B(x)) &= \frac{2|(A \cap B)|(x)}{|A|(x) + |B|(x)} \\ &= \frac{2 \sum min[A(x), B(x)]}{\sum A(x) + \sum B(x)} \end{aligned}$$

Finally, we can calculate the semantic similarity between the query and each of the documents.

$$d_f(Q, D_1) = 1.000 \quad (\text{'book'}, \text{'book'}) \tag{1}$$
$$d_f(Q, D_2) = 0.000 \quad (\text{'book'}, \text{'fantasy'}) \tag{2}$$
$$d_f(Q, D_3) = 0.400 \quad (\text{'book'}, \text{'hardcover'}) \tag{3}$$
$$d_f(Q, D_4) = 0.666 \quad (\text{'book'}, \text{'fantasy book'}) \tag{4}$$

TABLE I
OWLS-TC 3REV1 DATASET STATISTICS AFTER PREPROCESSING

Services	999
Tokens	388,586
Types (unique tokens)	11,222
Type/token ratio	0.33
Average types per service	11.2
Queries	29
Tokens	167
Types (unique tokens)	163
Type/token ratio	0.98
Average types per query	5.6
Judgements	3,584
relevant	1,333
not relevant	2,251

TABLE II
PREPROCESSING STEPS AND TOOLS

Processing step	Processing component
Initial tokenization	DKPro Core GPL[6] StanfordSegmenter
Re-tokenization 1	DKPro Core ASL[7] CamelCaseTokenSegmenter
Re-tokenization 2	DKPro Core ASL PatternBasedTokenSegmenter using the split pattern `[\\/?!&%\"'#<>_=.:;]+`
Re-tokenization 3	DKPro Core ASL TokenTrimmer removing leading and trailing dashes
Part-of-Speech tagging & lemmatization	DKPro Core ASL TreeTagger wrapper using the model for English
Stop word removal	DKPro Core ASL StopwordRemover using Snowball stop word list for English extended for XML Schema type names
Stemming	DKPro Core ASL Snowball stemmer[8]

The measure correctly calculates similarity for the trivial cases of self-comparison (1) as 1.0 and comparison between non-related concepts (2) as 0.0. The reason for (4) to yield a higher similarity than (3) is that D_4 uses the same level of abstraction as the query, both referring to the concept *book* while D_3 uses a more specific level of abstraction using the concept *hardcover*. This illustrates that the measure assigns a higher score if the query and a document refer to a concept at the same level of abstraction.

IV. EXPERIMENTS

A. Dataset

For our experiments, we used the OWLS-TC benchmark.[3] Since – to the best of our knowledge – no standard benchmark for service retrieval with natural language service requests exists, using OWLS-TC seems to provide the best means for experimental results. This benchmark has been used in previous related work by [4], [8], and it is used as a shared dataset in the annual Semantic Service Selection Contest[4] held since 2007. The current version OWLS-TC 3rev1 includes the WSDL documents we use for our experiments.

In total, OWL-S[5] and WSDL descriptions for 999 services are included in the dataset. We use only the WSDL descriptions in our experiments. Each service has a single operation with several input and output parameters. No documentation is included in any of the WSDL service descriptions. After preprocessing, this yields a corpus of $388,586$ tokens and $11,222$ unique tokens. The dataset contains 29 service descriptions serving as queries. We use the description field from the OWL-S version of these service descriptions as queries, e.g.:

- *This service returns lecturer of a university*
- *This service finds a comedy film for a title*
- *This service returns the destination where both games hiking and surfing are available*

Finally, the dataset contains $3,584$ relevance judgements which we use for our evaluation. Table I gives a detailed overview of the dataset characteristics.

[3]http://www.semwebcentral.org/frs/?group_id=89
[4]http://www-ags.dfki.uni-sb.de/~klusch/s3/
[5]http://www.w3.org/Submission/OWL-S

B. Evaluation Criteria

We evaluate the measures based on the following metrics:

Mean Average Precision (MAP): The *precision* $P(q)$ is the ratio of relevant documents $S(q)$ in the set of retrieved documents $T(q)$ for a query q from the query set Q:

$$P(q) \quad = \quad \frac{|S(q) \cap T(q)|}{|T(q)|}$$

The *average precision* $AP(q)$ calculates the sum over the *precision-at-rank* $P(q, r)$, if the document at rank r is relevant, divided by the number of relevant documents. $P(q, r)$ takes only retrieved documents up to rank r into account. Whether a document is relevant to the query is determined by the relevance function $r : Q \times D \rightarrow \{0, 1\}$. $T_r(q)$ is the retrieved document at rank r.

$$AP(q) \quad = \quad \frac{\sum_{r=1}^{|T(q)|}(P(q, r) * r(q, T_r(q)))}{|S(q)|}$$

The *mean average precision* (MAP) is calculated from the average precision for each query:

$$MAP \quad = \quad \frac{\sum_{q \in Q}(AP(q))}{|Q|}$$

Missed relevant documents: The number of relevant documents that were not retrieved is an indicator for the impact of the vocabulary gap on the retrieval method in our experiments. The number is calculated as the sum of relevant documents missed over all queries and, thus, is a fraction of the total number of documents judged as relevant.

C. Experimental Setup

We perform tokenization, lemmatization, stop-word removal, and stemming using components from the Darmstadt Knowledge Processing (DKPro) software repository.[9] More detailed information is given in table II.

[6]http://dkpro-core-gpl.googlecode.com
[7]http://dkpro-core-asl.googlecode.com
[8]http://snowball.tartarus.org/
[9]http://www.ukp.tu-darmstadt.de/research/projects/dkpro/

To calculate similarity, we use the following methods and implementations:

- bag-of-word based VSM implemented by the Apache Lucene project;[10]
- ESA [10] as implemented by [18];
- LSA [9] as implemented by the S-Space project[11] modified to follow our tokenization scheme;
- the semantic term similarity measures developed by Resnik [13], Lin [14], and Jiang & Conrath [15].

To aggregate pair-wise term similarities, we use

- Kuhn-Munkres' algorithm [19] for weighted bi-partite graph matching as implemented by the TimeFinder[12] project to aggregate term similarity scores into document similarity scores;
- geometric mean, harmonic mean, max, mean and median calculation as implemented by Apache Commons Math.[13]

We use WordNet as a semantic resource for our semantic VSM as well as for the knowledge-based semantic similarity measures used for comparison. The knowledge-based approaches work with WordNet and thus require a lemmatized representation of queries and documents. We chose to use the same lemmatized representation for the corpus-based approaches as well for better comparability of results.

ESA is a corpus-based approach using term co-occurrence frequencies instead of hypernym/hyponym relations. We created two models for ESA. One model is based on WordNet, treating each synset as a concept, and its gloss (together with the example sentences) as the concept's textual representation. The other model is based on Wiktionary glosses. Using Wiktionary for the computation of semantic relatedness has shown good results in the past [20]. Since glosses are often missing in Wiktionary, we also construct pseudo glosses by concatenating concepts that are in close relation (synonymy, hypernymy, meronymy, etc.) to the original concept as proposed in [21].

We apply LSA directly to the service descriptions from the OWLS-TC dataset. In their evaluation of LSA, Landauer et al. [9] observe the best results with a 300-dimensional concept space on a corpus with around 60,000 unique words. Given the comparatively small number of unique tokens (approx. 11,000, cf. table I) in the service descriptions, we used a concept space of only 100 dimensions.

V. Results

A. Aggregated term-level semantic similarity measures

In our first experiment, we calculate semantic similarity at the term-level and use different aggregation strategies to aggregate term-level scores to document-level scores. We use MAP as an evaluation measure. Table III compares the results obtained from the measures by Resnik [13], Lin [14], and Jiang & Conrath [15] as well as KB-FSM, ESA, and LSA.

[10]http://lucene.apache.org/

[11]http://code.google.com/p/airhead-research/

[12]http://timefinder.sourceforge.net/

[13]http://commons.apache.org/math/

We observe that the bi-partite graph aggregation strategy yields the best MAP score for each measure. We attribute this to the fact, that it best reflects the particular nature of the retrieval task: a relevant service offer should include all features specified in the request, but it may well provide additional features. The bi-partite graph aggregation strategy allows for a perfect relevance score in such cases, because a score of 1.0 can be reached if all query terms appear in the document. This is also true for the *max* aggregation, but it over-generalizes as it effectively only compares one query term with one document term. The averaging aggregation strategies can only reach a perfect relevance score if query and document are actually identical.

It is interesting to observe that KB-FSM outperforms Lin, Jiang & Conrath and Resnik, which are all also based on information content and exploit the hierarchical structure of the semantic resource.

The best-performing measures in this experiment are, however, the corpus-based measures ESA and LSA. We see here that LSA performs best when trained on the dataset itself, and ESA performs best when trained on Wiktionary, so in the following experiments, we only use these configurations.

B. Document-level semantic similarity measures

In our second experiment, we compare the document-level semantic similarity measures to each other. The results of this experiment are given in the column s_{sem} of table IV. Here, we observe LSA to perform best, followed by KB-FSM and ESA. Internally, all document-level measures aggregate term-level semantic vectors into a document-level semantic vector which are then combined using an inner product to produce the relevance score. For LSA, this seems to work well. It shows the same performance as when used with bi-partite graph aggregation in the previous experiment. The performance of ESA and KB-FSM drops compared to the first experiment, so here the vector aggregation strategy or the inner product are problematic.

C. Combined measures

For our third experiment, we combine the similarity measures s_{sem} with the bag-of-word based VSM implemented by Apache Lucene s_{bow} to calculate the similarity between the service request Q and the service offer D. The parameter w controls the balance between the semantic and the bag-of-word based components of the combined measure.

$$s(Q, D, w) = w * s_{sem}(Q, D) + (1 - w) * s_{bow}(Q, D)$$

The results for s_{bow} alone can be found in the leftmost column of table IV – this corresponds to choosing $w = 0.0$. Comparing these results with the results of our first two experiments, we notice that s_{bow} alone already outperforms all semantic similarity measures (rightmost column, $w = 1.0$) except LSA. By combining s_{bow} with any of the semantic similarity measures (columns $0.1 \leq w \leq 0.9$), we can improve the results. Again, the combination with LSA works best.

TABLE III
MEAN AVERAGE PRECISION (MAP) USING DIFFERENT STRATEGIES OF AGGREGATING PAIR-WISE TERM SIMILARITY

	semantic resource	inner product	bi-partite	mean	max	median	geometric mean	harmonic mean
LSA (d=100)	OWLS-TC	Cosine	**.68**	.57	.37	.51	.35	.24
LSA (d=300)	WordNet	Cosine	**.59**	.41	.34	.12	.05	.03
ESA	Wiktionary	Avg. Prod.	**.65**	.59	.37	.31	.08	.07
ESA	WordNet	Avg. Prod.	**.64**	.58	.37	.24	.01	.01
KB-FSM	WordNet	Dice	**.64**	.46	.36	.10	.04	.02
Resnik	WordNet	-	**.53**	.43	.32	.17	.12	.08
Jiang & Conrath	WordNet	-	**.45**	.22	.32	.14	.15	.13
Lin	WordNet	-	**.48**	.38	.32	.18	.12	.08

D. Modified Dice coefficient

While analyzing the results of this experiment, we noticed that KB-FSM had problems producing good results as the number of terms in the service description increased. We found the reason, again, in the particular nature of the service retrieval task, as described in section I-B. The Dice coefficient used by KB-FSM cannot produce a perfect score unless the query and document term sets are identical. This means, the similarity score drops quickly if an offer provides more features than asked for in the request.

Thus, we conducted a fourth experiment in which we used a modified Dice coefficient d_{fl} that focusses on a large intersection between request and offer, and reduces the penalty when an offer provides more features than requested. When B is equal to or a subset of A, d_{fl} is equal to d_f (cf. section III). If B is a superset of A, its cardinality is artificially reduced compared to the actual Dice coefficient.

$$
\begin{aligned}
m &= max(|A|(x), |B|(x)) \\
w &= \frac{|A|(x)}{m} \\
d_{fl}(A(x), B(x)) &= \frac{(1+w)|(A \cap B)|(x)}{|A|(x) + wm} \\
&= \frac{(1+w)\sum min[A(x), B(x)]}{\sum A(x) + wm}
\end{aligned}
$$

As ESA and LSA are based on vectors representing sets, we could also use the modified Dice coefficent here as the inner vector product, trying to adapt them as well to the service retrieval task. Detailed results for this experiment are given in table V.

Using the modified coefficient, KB-FSM gets back to the level of performance it showed as a term-level similarity measure aggregated using the bi-partite graph strategy, which also reflects the particular nature of the task. For ESA, we can see a slight improvement when used stand-alone, but an overall drop in combination with Lucene. LSA consistently performs worse with this inner product. The reason for this should lie in the nature of the semantic vectors used by the different measures. The semantic vectors of KB-FSM are of variable size, depending on the number of concepts appearing in the query and in the document. For ESA and LSA, the vectors have a fixed size. The size of the ESA vectors depends on the number of documents in the training corpus, while the size of LSA vectors is an arbitrary parameter set during training. Therefore, those approaches are less susceptible to length differences between query and document and do not benefit from the modified coefficient.

E. Vocabulary gap

In our last experiment, we examine whether the semantic approaches were able to reduce or eliminate the vocabulary gap. As said initially, when the query and document use different vocabularies, bag-of-words based retrieval methods lead to unsatisfactory results, because they require the same terms to exist in both. For this experiment, we compare the bag-of-word based VSM s_{bow} with the best-performing configurations of ESA, LSA and KB-FSM from the previous experiments. We examined how many of the documents judged as relevant were not retrieved by the methods. The results are given in table VI. As expected, s_{bow} alone performs worst. When using the combined measures (cf. section V-C), the vocabulary gap effect is reduced or even eliminated. As LSA is trained on the corpus itself, it cannot find a document if the query uses only out-of-corpus vocabulary. This does not apply to ESA and KB-FSM which use a external knowledge sources that obviously has a good coverage of both, corpus and query terms.

VI. CONCLUSION

We have proposed KB-FSM, a computationally efficient knowledge-based semantic retrieval model based on fuzzy sets that combines the computational efficiency of a VSM with the level of control offered by a knowledge-based approach. Our model was evaluated in the context of service retrieval along with a set of some of the most widely used knowledge-based and corpus-based semantic retrieval models. We show that combinations of the semantic retrieval models with a bag-of-words based VSM can yield better results than either semantic or bag-of-words based models alone. In a comparative evaluation, KB-FSM yielded the best result amongst the knowledge-based approaches, performing similar to the best corpus-based approach.

We suggest considering either LSA or KB-FSM for service retrieval, depending if a sufficiently large training corpus is available or if a knowledge base with explicit semantic relations can be utilized. Both semantic approaches can benefit from being combined with a standard boolean VSM. While ESA has comparatively good results, it does not scale well

TABLE IV
MAP FOR DOCUMENT SIMILARITY MEASURE COMBINED WITH BAG-OF-WORD BASED VSM

			s_{bow}	w =									s_{sem}
	resource	inner product	0.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	1.0
LSA (d=100)	OWLS-TC	cos	.66	.69	**.70**	**.70**	**.70**	**.70**	**.70**	**.70**	.69	.69	.68
ESA	Wiktionary	$avg.product$	.66	**.69**	**.69**	**.69**	**.69**	**.69**	**.69**	**.69**	**.69**	**.69**	.56
KB-FSM	WordNet	d_f	.66	.69	**.70**	.69	.69	.69	.68	.67	.65	.63	.60

TABLE V
MAP FOR SIMILARITY MEASURES USING MODIFIED DICE d_{fl} COMBINED WITH BAG-OF-WORD BASED VSM

			s_{bow}	w =									s_{sem}
	resource	inner product	0.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	1.0
LSA (d=100)	OWLS-TC	d_{fl}	.66	.68	.68	.68	**.69**	**.69**	**.69**	.68	.67	.67	.64
ESA	Wiktionary	d_{fl}	.66	**.68**	**.68**	**.68**	.67	.67	.66	.64	.62	.60	.58
KB-FSM	WordNet	d_{fl}	.66	.69	**.70**	**.70**	**.70**	**.70**	.69	.68	.67	.66	.64

TABLE VI
RELEVANT DOCUMENTS NOT FOUND

	semantic resource	inner product / aggregation	w	missed rel. doc.
s_{bow}	-	-	-	10% (127)
LSA (d=100)	OWLS-TC	cos	.4	2% (31)
ESA	Wiktionary	$avg.product$	.8	$< 1\%$ (1)
KB-FSM	WordNet	d_{fl}	.3	none

to larger document collections. We do not recommend using aggregated path-based similarity measures, as they produce inferior results and do not scale well to larger documents.

In further work, we plan to evaluate KB-FSM in other IR scenarios using different datasets and knowledge bases. We would like, in particular, to include domain-specific ontologies in addition to WordNet.

ACKNOWLEDGMENT

The project was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference "01MQ07012". The authors take the responsibility for the contents. The information in this document is proprietary to the following THESEUS consortium members funded by means of the German Federal Ministry of Economy and Technology: Technische Universität Darmstadt. The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law.

REFERENCES

[1] B. Caprile and P. Tonella, "Nomen est omen: Analyzing the language of function identifiers," in *WCRE '99: Proceedings of the Sixth Working Conference on Reverse Engineering*. Washington, DC, USA: IEEE Computer Society, 1999, p. 112.

[2] E. Stroulia and Y. Wang, "Structural and semantic matching for assessing web-service similarity," *International Journal of Cooperative Information Systems*, vol. 14, pp. 407–437, 2005.

[3] X. Dong, A. Halevy, J. Madhavan, E. Nemes, and J. Zhang, "Similarity search for web services," in *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*. VLDB Endowment, 2004, pp. 372–383.

[4] P. Plebani and B. Pernici, "URBE: Web service retrieval based on similarity evaluation," *IEEE Trans. on Knowl. and Data Eng.*, vol. 21, no. 11, pp. 1629–1642, 2009.

[5] M. Klusch, P. Kapahnke, and I. Zinnikus, "Hybrid adaptive web service selection with SAWSDL-MX and WSDL-Analyzer," in *ESWC 2009 Heraklion: Proceedings of the 6th European Semantic Web Conference on The Semantic Web*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 550–564.

[6] G. Salton, *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[7] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*, 1st ed. Cambridge University Press, July 2008.

[8] M. Klusch, B. Fries, and K. Sycara, "Automated semantic web service discovery with OWLS-MX," in *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM, 2006, pp. 915–922.

[9] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse Processes*, no. 25, pp. 259–284, 1998.

[10] E. Gabrilovich and S. Markovitch, "Computing semantic relatedness using wikipedia-based explicit semantic analysis," in *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, pp. 1606–1611.

[11] M. Lesk, "Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone," in *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*. New York, NY, USA: ACM, 1986, pp. 24–26.

[12] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1994, pp. 133–138.

[13] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy," in *IJCAI'95: Proceedings of the 14th international joint conference on Artificial intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 448–453.

[14] D. Lin, "An information-theoretic definition of similarity," in *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 296–304.

[15] J. J. Jiang and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy," *CoRR*, vol. cmp-lg/9709008, 1997.

[16] N. Seco, T. Veale, and J. Hayes, "An intrinsic information content metric for semantic similarity in WordNet," in *ECAI*, R. L. de Mántaras and L. Saitta, Eds., 2004, pp. IOS Press–1090.

[17] L. Zadeh, "Fuzzy sets," *Information Control*, vol. 8, pp. 338–353, 1965.

[18] G. Szarvas, T. Zesch, and I. Gurevych, "Combining heterogeneous knowledge resources for improved distributional semantic models," in *Proceedings of CICLing 2011*, Feb 2011.

[19] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistic Quarterly*, vol. 2, pp. 83–97, 1955.

[20] T. Zesch, C. Müller, and I. Gurevych, "Using Wiktionary for computing semantic relatedness," in *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, Chicago, IL, USA, Jul 2008, pp. 861–867.

[21] I. Gurevych, "Using the structure of a conceptual network in computing semantic relatedness," in *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP)*, Jeju Island, Republic of Korea, Oct 2005, pp. 767–778.