Learning Semantic Query Suggestions

Edgar Meij[1], Marc Bron[1], Laura Hollink[2], Bouke Huurnink[1], and Maarten de Rijke[1]

[1] ISLA, University of Amsterdam, Science Park 107, 1098 XG Amsterdam
{edgar.meij,m.m.bron,bhuurnink}@uva.nl, mdr@science.uva.nl
[2] Dept. of Computer Science, VU University Amsterdam,
de Boelelaan 1081a, 1081 AH Amsterdam
hollink@cs.vu.nl

Abstract. An important application of semantic web technology is recognizing human-defined concepts in text. Query transformation is a strategy often used in search engines to derive queries that are able to return more useful search results than the original query and most popular search engines provide facilities that let users complete, specify, or reformulate their queries. We study the problem of *semantic query suggestion*, a special type of query transformation based on identifying semantic concepts contained in user queries. We use a feature-based approach in conjunction with supervised machine learning, augmenting term-based features with search history-based and concept-specific features. We apply our method to the task of linking queries from real-world query logs (the transaction logs of the Netherlands Institute for Sound and Vision) to the DBpedia knowledge base. We evaluate the utility of different machine learning algorithms, features, and feature types in identifying semantic concepts using a manually developed test bed and show significant improvements over an already high baseline. The resources developed for this paper, i.e., queries, human assessments, and extracted features, are available for download.

1 Introduction

Human-defined concepts are fundamental building blocks of the semantic web. When used as annotations for documents or text fragments they can provide explicit anchoring in background knowledge and enable intelligent search and browsing facilities. As such, an important application of ontological knowledge is augmenting unstructured text with links to relevant, human-defined concepts. For the author or reader of the text, this augmentation may supply useful pointers, for example to the concepts themselves or to other concepts related to the ones found. For ontology learning applications, such links may be used to learn new concepts or relations between them [34]. Recently, data-driven methods have been proposed to generate links between phrases appearing in full-text documents and a set of ontological concepts known a priori. Mihalcea and Csomai [24] propose the use of several linguistic features in a machine learning framework to link phrases in full-text documents to Wikipedia articles and this approach is further improved upon by Milne and Witten [25]. Because of the connection between Wikipedia and DBpedia, such data-driven linking methods help us establish links between textual documents and Linked Open Data [2, 3, 9, 33].

Another, more challenging instantiation of linking text to human-defined concepts in a knowledge source is *semantic query suggestion*. Query suggestion is a strategy to

derive terms that are able to return more relevant results than the initial query. Commonly used approaches to query suggestion (sometimes referred to as a form of query expansion) are highly data-driven and based mostly on term frequencies [21, Chapter 9]. *Semantic* query suggestion, in contrast, tries to understand (or learn) which concepts the user used in her query or, phrased alternatively, the concepts she is interested in and wants to find.[1] Moreover, the properties of each concept, and any other resources associated with it, could serve as additional, useful information for the user. In our current work, we use DBpedia as our target ontology. As an example of our task, consider the query "obama white house". A semantic query suggestion algorithm should return suggestions in the form of the (DBpedia) instances labeled "Barack Obama" and "White House". Identifying such semantic suggestions serves multiple purposes: it can (i) help the user acquire contextual information, (ii) suggest related concepts or associated terms that may be used for search, and (iii) provide valuable navigational suggestions.

In this paper we address the semantic query suggestion task and automatically link queries to DBpedia concepts. We propose a novel method that leverages the textual representation of each concept as well as query-based and concept-based features. Working with user queries, which are typically quite short [31], implies that we cannot use previously established approaches that rely on textual context such as shallow parsing or part-of-speech tagging [24]. One interesting aspect of working with user queries is that we have history-based information available (from the session or the user) that can potentially be used to help address the semantic query suggestion task.

Our approach proceeds as follows. First, we use language modeling for information retrieval (IR) techniques to retrieve the most relevant concepts for the full query and for each n-gram (i.e., contiguous sequence of n words) in the query. Second, we use supervised machine learning methods to decide which of the retrieved concepts should be kept and which should be discarded. In order to train the machine learner, we examined close to 1000 queries from a search engine for a digital multimedia archive and manually linked over 600 of these to relevant concepts in DBpedia.[2] The research questions we address are the following.

- Can we cast semantic query suggestion as a ranking problem?
- What is the best way of handling the terms in the input query?
- Can we use features pertaining to the query terms, concepts, and history to improve upon a basic retrieval based approach?
- Which type of the feature helps most? Which separate feature is most informative?

Our main contribution is threefold: we introduce the problem of semantic query suggestion, provide a very effective approach to the problem plus an analysis of the contributions of the features used, and we make available resources to enable future work on the problem. The remainder of this paper is structured as follows. In Section 2 we discuss related work. Sections 3 and 4 detail the semantic query suggestion task and our approach. Our experimental setup is described in Section 5 and our results are presented in Section 6. We end with a concluding section.

[1] We use "ontology" to refer to the full spectrum of conceptualizations, ranging from glossaries to formal ontologies [22]. We refer to an instance in DBpedia as "concept" [33].

[2] The queries, human assessments, and extracted features are publicly available for download at `http://ilps.science.uva.nl/resources/iswc09_annotations`.

2 Related Work

Linking terms or phrases to ontologies is related to several areas of research. These include semantic web areas such as ontology matching and semantic annotation, but also areas from language technology and information retrieval.

In *ontology matching* relations between concepts from different ontologies are found. These relations are based on a comparison of instances, concept labels, semantic structure, or ontological features such as constraints or properties, sometimes exploiting auxiliary external resources such as the lexical resource WordNet or the upper ontology DOLCE [30]. E.g., Wang et al. [36] develop a machine learning technique to learn the relationship between the similarity of instances and the validity of mappings between concepts. Other approaches are designed for lexical comparison of concept labels in the source and target ontology and do not use semantic structure nor instances (e.g. [32]). This type of matching is sometimes referred to as 'lexical matching' and is used in cases where the ontologies do not have any instances or structure; e.g., in Aleksovski et al. [1] lexical comparison of labels is used to map both the source and the target ontology to a semantically rich external source of background knowledge.

Lexical matching is very similar to our task at hand, as we do not have any semantic structure in the queries. Since 2008, the Ontology Alignment Evaluation Initiative has contained a task similar to ours, where participants link a largely unstructured thesaurus, the GTAA, to DBpedia [10]. Our approach is different, however, in two ways. First, we use the interaction history with the system in the form of user sessions to obtain a certain amount of contextual information. Second, the fact that the source terms that we are trying to link are natural language captured in user queries instead of standardized concept labels makes the task intrinsically harder.

A lot of work has been done on semantic annotation: the process of relating documents to concepts from ontologies or other sources of structured knowledge, such as Wikipedia. Many restricted forms of the problem have been addressed. The detection of named entities in pieces of text is an important sub-problem of semantic annotation. For example, Kiryakov et al. [20] create links between named entities in text and concepts from a light-weight ontology. Bhole et al. [8] propose an interesting example of semantic document analysis, where named entities are related over time using Wikipedia. Chemudugunta et al. [11] do not restrict themselves to named entities, but instead link all words in a document to ontological concepts. They use latent topic models to learn links between words and concepts. Other sub-problems of semantic annotation include sense tagging and word sense disambiguation [13]. Some of the techniques developed there have fed into automatic link generation between full-text documents and Wikipedia. For example, Milne and Witten [25] depend heavily on contextual information (terms and phrases) around the source text to determine the best Wikipedia articles to link to (like typical named entity recognition and sense tagging methods). This work was based on earlier work by Mihalcea and Csomai [24]. Similarly, we also consider a two-step approach to linking text to Wikipedia/DBpedia; first keyword extraction is performedi, and next, each extracted keyword is linked to a Wikipedia article. The authors apply part-of-speech tagging and develop several ranking procedures for candidate Wikipedia articles. Our approach differs in that we do not limit ourselves to exact matches with the query terms. Another distinct difference between our approach

and those mentioned above is that while they work with full-text documents, we utilize much sparser data in the form of user queries. As such we can not easily use techniques such as part-of-speech tagging or lean too heavily on context words for disambiguation. As will be detailed below, our approach uses session history and n-grams in the queries to obtain contextual information instead.

Turning to semantic query analysis (as opposed to semantic analysis of full documents), Guo et al. [14] perform named entity recognition in queries; they recognize a single entity in each query and subsequently classify it into one of a small set of predefined classes such as "movie" or "video game". We do not impose the restriction of having a single concept per query and, furthermore, our list of candidate concepts is much larger, i.e., all articles in Wikipedia. Several other approaches have been proposed that link queries to a limited set of categories. Mishne and de Rijke [26] use online product search engines to link queries to product categories; Beitzel et al. [5] link millions of queries to 17 topical categories based on a list of manually pre-categorized queries; Jansen et al. [16] use commonly occurring multimedia terms to categorize audio, video, and image queries.

As to related work on query suggestions, some approaches use query logs to determine which queries or which rewrites occur frequently [17]. Others perform query analysis and try to identify the most relevant terms [6], to predict the query's performance a priori [40], or combine the two [7]. Bendersky and Croft [6] use part-of-speech tagging and a supervised machine learning technique to identify the "key noun phrases" in natural language queries. Key noun phrases are phrases which convey the most information in a query and contribute most to the resulting retrieval performance; we evaluate several of the features proposed by these authors.

3 The Task

The semantic query suggestion task we address in this paper is the following. Given a query that is submitted to a search engine, identify the relevant concepts that the user entered in her query where the concepts are taken from an existing knowledge base or ontology. We address our task in the setting of a digital archive, specifically of the Netherlands Institute for Sound and Vision ("Sound and Vision"). Sound and Vision maintains a large digital audiovisual collection, currently containing over a million objects and daily updated with new broadcasts. Users of the archive consist primarily of media professionals who use an online search interface to locate audiovisual items to be used in new programs such as documentaries and news reviews.

Sound and Vision is in the process of linking all its digital resources to a variety of structured knowledge sources. Because of its central role in the Linked Open Data initiative, our knowledge source of choice for semantic query suggestion is DBpedia. Thus, in practical terms, the task we are facing is: given a query (within a session, for a given user), produce a ranked list of concepts from DBpedia that are mentioned or meant in the query. These concepts could then be used to suggest relevant multimedia items associated with each concept or to suggest contextual information, such as text snippets from the Wikipedia article.

4 Approach

Our approach to suggesting DBpedia concepts for user queries consists of two stages. In the first stage, a ranked list of possible concepts for the query is generated using a language modeling framework (cf. Section 4.1). To create this ranking we consider two approaches; one approach is to extract all n-grams in the query and generate a ranking for each. The other approach is to use the entire query to create a single concept ranking. We use various textual representations of each DBpedia concept, including the accompanying Wikipedia article text, its label, and the text used in the hyperlinks pointing to it. An example of the first stage of ranking concepts is provided in Table 1 for the query "obama white house". From the example it is clear why we consider these two approaches. Should we simply use the full query on its own (first row), we would miss the relevant concept "Barack Obama". However, as can be seen from the last two rows, considering all n-grams also introduces noise.

Table 1. An example of generating n-grams for the query "obama white house" and retrieved candidate concepts, ranked by retrieval score. Correct concepts in boldface.

N-gram (Q)	Candidate concepts
obama white house	**White House**; White House Station; President Coolidge; Sensation White
obama white	Michelle Obama; **Barack Obama**; Democratic Pre-elections 2008; January 17
white house	**White House**; White House Station; Sensation; President Coolidge
obama	**Barack Obama**; Michelle Obama; Presidential Elections 2008; Hillary Clinton
white	Colonel White; Edward White; White County; White Plains Road Line
house	House; Royal Opera House; Sydney Opera House; Full House

In the second stage supervised machine learning is used to decide which of the candidate concepts in the ranked lists should be kept as viable concepts for the query in question. In order to train these machine learning algorithms, we asked annotators to assess queries submitted to Sound and Vision and manually link them to relevant DBpedia concepts. More details with respect to the test collection and the manual annotations are listed in Section 5. The machine learning algorithms we consider are Naive Bayes, Decision Trees, and Support Vector Machines [35, 37] and are introduced in Section 4.2. As input to the machine learning algorithms we extract a set of features related to the query n-gram, concept, and the session in which the query appears as detailed in Section 4.3.

4.1 Ranking Concepts

We base our concept ranking framework within the language modeling paradigm, as it is a theoretically transparent retrieval approach that is competitive in terms of retrieval effectiveness [15, 28, 39]. Here, a query is viewed as having been generated from an underlying document language model, where some words are more probable to occur than others. At retrieval time each document is scored according to the estimated likelihood that the words in the query were generated by a random sample of the language model underlying the document. These word probabilities are generally estimated from the document itself (using maximum likelihood estimation) and combined with back-

ground collection statistics to overcome zero probability and data sparsity issues; a process known as *smoothing*.

For the n-gram based re-ranking, we extract all n-grams from each query $\mathbf{Q}$ (where $1 \leq n \leq |\mathbf{Q}|$) and create a ranked list of concepts for each individual n-gram, Q. For the full-query based reranking approach, we use the same method but add the additional constraint that $n = |\mathbf{Q}|$. The problem of ranking DBpedia concepts given an n-gram can then be formulated as follows. Each concept c should be ranked according to the probability that it was generated by the n-gram $P(c|Q)$, which can be rewritten using Bayes' rule as:

$$P(c|Q) = \frac{P(Q|c)P(c)}{P(Q)}.$$

Here, the term $P(Q)$ is the same for all concepts and be ignored for ranking purposes. The term $P(c)$ indicates the prior probability of selecting a concept, which we assume to be uniform. Assuming independence between the individual terms $q \in Q$, as is common in the field of information retrieval [4], we obtain

$$P(c|Q) \propto P(c) \prod_{q \in Q} P(q|c)^{n(q,Q)}, \tag{1}$$

where $n(q, Q)$ indicates the count of term q in n-gram Q. The probability $P(q|c)$ is smoothed using Bayes smoothing with a Dirichlet prior [39], which is formulated as:

$$P(q|c) = \frac{n(q, c) + \mu P(q)}{\sum_q n(q, c) + \mu}, \tag{2}$$

where $P(q)$ indicates the probability of observing q in a large background collection and μ is a hyperparameter that controls the influence of the background corpus.

Since each DBpedia concept is linked to its Wikipedia counterpart, we can use the textual representations of the associated wikipedia pages for retrieval. In particular, we perform retrieval using the title of the article, the content, and the text used for the hyperlinks pointing to it from other Wikipedia articles.

4.2 Learning to Rerank Concepts

Once we have obtained a ranked list of possible concepts for each n-gram in the query, we turn to concept selection. In this stage we need to decide which of the candidate concepts are most viable. We use a supervised machine learning approach, which takes as input a set of labeled examples (query to concept mappings) and several features of these examples (detailed below). We choose to compare a Naive Bayes (NB) classifier, with a Support Vector Machine (SVM) classifier and a decision tree classifier (J48)—a set representative of the state-of-the-art in classification. We experiment with multiple classifiers in order to confirm that our results are generally valid, i.e., not dependent on any machine learning algorithm.

4.3 Features Used

We employ several *types* of features, each associated with either the current query n-gram, the current concept, their combination, or the current search history. Unless indicated otherwise, we consider Q to be a phrase when determining the features.

Table 2. Features used, grouped by type.

N-gram features	
$LEN(Q) = \|Q\|$	Number of terms in the phrase Q
$IDF(Q)$	Inverse document frequency of Q
$WIG(Q)$	Weighted information gain using top-5 retrieved concepts
$QE(Q)$	Number of times Q appeared as *whole* query in query log
$QP(Q)$	Number of times Q appeared as *partial* query in query log
$QEQP(Q)$	Ratio between QE and QP
$SNIL(Q)$	Does a sub-n-gram of Q fully match with any concept label?
$SNCL(Q)$	Is a sub-n-gram of Q contained in any concept label?
Concept features	
$INLINKS(c)$	The number of concepts linking to c
$OUTLINKS(c)$	The number of concepts linking from c
$GEN(c)$	Function of depth of c in SKOS category hierarchy [25]
$CAT(c)$	Number of associated categories
$REDIRECT(c)$	Number of redirect pages linking to c
N-gram + concept features	
$TF(c,Q) = \frac{n(Q,c)}{\|c\|}$	Relative phrase frequency of Q in c, normalized by length of c
$TF_f(c,Q) = \frac{n(Q,c,f)}{\|f\|}$	Relative phrase frequency of Q in representation f of c, normalized by length of f
$POS_n(c,Q) = pos_n(Q)/\|c\|$	Position of nth occurrence of Q in c, normalized by length of c
$SPR(c,Q)$	Spread (distance between the last and first occurrences of Q in c)
$TF \cdot IDF(c,Q)$	The importance of Q for c
$RIDF(c,Q)$	Residual IDF (difference between expected and observed IDF)
$\chi^2(c,Q)$	χ^2 test of independence between Q in c and in collection $Coll$
$QCT(c,Q)$	Does q contain the label of c?
$TCQ(c,Q)$	Does label of c contain q?
$TEQ(c,Q)$	Does label of c equal q?
$SCORE(c,Q)$	Retrieval score of c w.r.t Q
$RANK(c,Q)$	Retrieval rank of c w.r.t Q
History features	
$CCIH(c)$	Number of occurrences of label of c appears as query in history
$CCCH(c)$	Number of occurrences of label of c appears in any query in history
$CIHH(c)$	Number of times c is retrieved as result for any query in history
$CCIHH(c)$	Number of times label of c equals title of any result for any query in history
$CCCHH(c)$	Number of times title of any result for any query in history contains label of c
$QCIHH(Q)$	Number of times title of any result for any query in history equals Q
$QCCHH(Q)$	Number of times title of any result for any query in history contains Q
$QCIH(Q)$	Number of times Q appears as query in history
$QCCH(Q)$	Number of times Q appears in any query in history

N-gram Features These features are based on information from an n-gram and are listed in Table 2 (first group). $IDF(Q)$ indicates the relative number of concepts in which Q occurs, which is defined as $IDF(Q) = \log(|Coll|/df(Q))$, where $|Coll|$ indicates the number of documents in the collection and $df(Q)$ the number of documents in which Q occurs [4]. $WIG(Q)$ indicates the weighted information gain, that was proposed by Zhou and Croft [40] as a predictor of the retrieval performance of Q. It is the only feature which uses the set of all candidate concepts retrieved for this n-gram, C_Q, and determines the relative probability of Q occurring in these documents as compared to the collection. Formally:

$$WIG(Q) = \frac{\frac{1}{|C_Q|}\sum_{c \in C_Q} \log(P(Q|c)) - \log(P(Q))}{\log P(Q)}.$$

$QE(Q)$ and $QP(Q)$ indicate the number of times this n-gram appears in the entire query logs as a whole or partial query respectively.

Concept Features Table 2 (second group) lists the features related to a DBpedia concept. This set of features is related to the knowledge we have of the current candidate concept, such as the number of other concepts linking to or from it, the number of associated categories (the count of the DBpedia property `skos:subject`), and the number of redirect pages pointing to it (the DBpedia property `dbpprop:redirect`).

N-gram + Concept Features This set of features considers an n-gram and a concept (Table 2, third group). We consider the relative frequency of occurrence of the n-gram as a phrase in the corresponding Wikipedia article, in the separate document representations (title, content, anchor texts, first sentence, and first paragraph of the Wikipedia article), the position of the first occurrence of the n-gram, the distance between the first and last occurrence, and various IR-based measures. Of these, $RIDF$ [12] is the difference between expected and observed IDF for a concept, which is defined as $RIDF(c, Q) = \log(|Coll|/df(Q)) + \log(1 - \exp(-n(Q, Coll)/|Coll|))$. We also consider whether the label of the concept, i.e. the Wikipedia article title, matches Q and we include the current retrieval information.

History Features Finally, we consider features based on the previous queries that were issued in the same session (Table 2, fourth group). These features look at either the current candidate concept or current n-gram and see whether they occur (partially) in the previous queries or in the retrieved candidate concepts.

Below, we compare the effectiveness of the features listed above for our semantic query suggestion task.

5 Experimental Setup

We introduce the experimental environment and the experiments that we perform to answer the research questions from Section 1. We also introduce our evaluation measures and describe our manual assessments, but start with detailing our data sets.

5.1 Data

Two main types of data are needed for our experiments: queries and DBpedia concepts. We have access to a set of 264,503 queries issued between 18 November 2008 to 15

May 2009 to Sound and Vision. Sound and Vision logs the actions of users on the site, generating session identifiers and time stamps. This allows a series of consecutive queries to be linked to a single search session, where a session is identified using a session cookie. A session is terminated once the user closes the browser. An example is given in Table 3. All queries were Dutch language queries; however, nothing in our semantic query suggestion approach is language dependent. As the "history" of a query, we took all queries previously issued in the same user session.

Table 3. An example of queries issued in a (partial) session, translated to English.

Session ID	Query ID	Query ($\mathbf{Q}$)
jyq4navmztg	715681456	santa claus canada
jyq4navmztg	715681569	santa claus emigrants
jyq4navmztg	715681598	santa claus australia
jyq4navmztg	715681633	christmas sun
jyq4navmztg	715681789	christmas australia
jyq4navmztg	715681896	christmas new zealand
jyq4navmztg	715681952	christmas overseas

The DBpedia version we use is the most recent Dutch version (3.2). We downloaded the Wikipedia dump from which this DBpedia version was created (dump date 20080609); this dump is used for all our text-based processing steps and features.

5.2 Training Data

For training and testing purposes, four assessors were asked to manually link 998 queries to DBpedia concepts. The assessors were presented with a list of sessions and the queries in them. Once a session had been selected, they were asked to find the most relevant DBpedia concepts given each query and its preceding session history if any. Our assessors were able to search through Wikipedia using the fields described in Section 4.1. Besides indicating relevant concepts, the assessors could also indicate whether a query was ambiguous, contained a typographical error, or whether they were unable to find any relevant concept. For our experiments, we removed all the assessed queries in these "anomalous" categories and were left with a total of 629 assessed queries in 193 sessions.[3] In this sample the average query length is 2.14 terms per query. Each query has 1.34 concepts annotated on average.

5.3 Parameters

As to retrieval, we use the entire Wikipedia document collection as background corpus and set μ to the average length of a Wikipedia article [39], i.e., $\mu = 315$ (cf. Eq. 2). For classification we use the following three machine learning algorithms in our experiments: J48, Naive Bayes and Support Vector Machines. The implementations are taken from the Weka machine learning toolkit [37]. J48 is a decision tree algorithm and the Weka implementation of C4.5 [29]. The Naive Bayes classifier uses the training data to estimate the probability that an instance belongs to the target class, given the presence

[3] We focus on evaluating the actual semantic query suggestions and discard queries which the assessors deemed too anomalous to confidently link to any concept.

of each feature. By assuming independence between the features these probabilities can be combined to calculate the probability of the target class given all features [19]. SVM uses a sequential minimal optimization algorithm for training with polynomial kernels as described in [27]. The training instances are represented as n-dimensional vectors and two decision hyperplanes are created that best separate the instances of the target classes. The distance between the hyperplanes is called the margin and by maximizing this distance the generalization error of the classifier is minimized. For all algorithms we do not perform extensive optimization of the parameter settings and use the default weka parameters.[4] Whether fine-grained parameter tuning is beneficial and, thus, whether our choice negatively influences the experimental outcomes is a topic for future work.

5.4 Testing and Evaluation

We define semantic query suggestion as a ranking problem. In this paper, the system has to return five concepts for a given input query; the assessments described above are used to determine the relevance of these five concepts. We employ several measures which are well-known in the field of information retrieval [4], namely precision@1 (P1; how many relevant concepts are retrieved at rank 1), r-precision (R-prec; precision@r where r equals the size of the set of known relevant concepts for this query), recall (which percentage of relevant concepts were retrieved?), mean reciprocal rank (MRR; the reciprocal of the rank of the first correct concept), and the success rate @5 (SR; a binary measure that indicates whether at least one correct concept has been returned in the top-5).

To verify the generality of the machine learning algorithms, we perform 10-fold cross validation [37], which reduces the possibility of errors being caused by artifacts in the data. The reported scores are averaged over all folds and all evaluation measures are averaged over the queries used for testing. For determining the statistical significance of the observed differences between the various runs we use one-way ANOVA to determine if there is a significant difference ($\alpha \leq 0.05$). We then use the Tukey-Kramer test to determine which of the individual pairs are significantly different. We designate the best result in each table in bold face.

6 Results

In this section we report on the experimental results which answer the research questions from Section 1. We compare three approaches to the semantic query suggestion task:

(i) a *baseline* which retrieves concepts based solely on their textual representation in the form of the associated Wikipedia article,
(ii) *n-gram based reranking* which extracts all n-grams from the query and uses machine learning to identify the best concepts, and
(iii) *full-query based reranking* which does not extract n-grams, but calculates feature values based on the full query.

After we have described the results for each approach, we zoom in on the most informative features and the specific feature types.

[4] See http://weka.sourceforge.net/doc.

6.1 Baseline

As our baseline, we take the entire query as issued by the user and employ Eq. 1 to rank DBpedia concepts based solely on their textual representation (this techniques is similar to using a search engine to perform a search within the Dutch Wikipedia). We consider two forms of textual representation: "content" and "full text". The former consists of the textual content of a Wikipedia article corresponding to a DBpedia concept, the latter adds to this the title of the article and the anchor texts of hypertext links in Wikipedia that point to the article at hand.

Table 4. Results of ranking concepts based on using the entire query **Q** and either the content of the Wikipedia article or the full text associated with each DBpedia concept.

	P1	R-prec	Recall	MRR	SR
full text	**0.5636**	**0.5216**	**0.6768**	**0.6400**	**0.7535**
content	0.5510	0.5134	0.6632	0.6252	0.7363

Table 4 shows the results of the baseline. From this table we observe that including the title and anchor texts of the incoming links results in improved retrieval performance overall. Notice that this is a strong baseline; on average, over 65% of the relevant concepts are correctly identified in the top-5 and, further, over 55% of the relevant concepts are retrieved at rank 1. The success rate indicates that for 75% of the queries at least one relevant concept is retrieved in the top-5.

Table 5. An example of baseline results for the n-grams in the query "challenger wubbo ockels", ranked by retrieval score. Concepts labeled as correct in boldface.

N-gram	Candidate concepts
challenger	**Space Shuttle Challenger**; Challenger; Bombardier Challenger; STS-61-A
wubbo	**Wubbo Ockels**; Spacelab; Canon of Groningen; Superbus; André Kuipers
ockels	**Wubbo Ockels**; Spacelab; Superbus; Canon of Groningen; André Kuipers
challenger wubbo	**Wubbo Ockels**; STS-61-A; **Space Shuttle Challenger**; Spacelab; STS-9
wubbo ockels	**Wubbo Ockels**; Spacelab; Superbus; Canon of Groningen; Andr Kuipers
challenger wubbo ockels	**Wubbo Ockels**; STS-61-A; Spacelab; **Space Shuttle Challenger**

6.2 N-gram based reranking

Table 5 shows the result for the baseline (last row) and the query "challenger wubbo ockels". As is clear from this example, the two relevant concepts are retrieved at ranks 1 and 4. When we look at the same results for the n-grams in the query, however, one of the relevant concepts is retrieved at the first position for each n-gram. This example and the one in Table 1 suggest that it will be beneficial to consider all possible n-grams in the query. In this section we report on the results of extracting n-grams from the query, generating features for each, and subsequently applying the machine learning algorithms to decide which of the suggested concepts to keep. The features used here are described in Section 4.2.

Table 6 shows the results of applying the machine learning algorithms on the extracted n-gram features. We note that J48 and SVM are able to improve upon the baseline re-

Table 6. Results for n-gram based reranking. ▲ ▼ and ° indicate that a score is significantly better, worse or statistically indistinguishable respectively. The leftmost symbol represents the difference with the baseline, the next with the J48 run, and the rightmost with the NB run.

Machine learner	P1	R-prec	Recall	MRR	SR
baseline	0.5636	0.5216	0.6768	0.6400	0.7535
J48	0.6510$^{\blacktriangle}$	0.5604°	0.7245°	0.7441$^{\blacktriangle}$	0.7958°
NB	0.4665$^{\blacktriangledown\circ}$	0.4344$^{\blacktriangledown\blacktriangledown}$	0.6984$^{\circ\circ}$	0.7100$^{\circ\circ}$	0.7614$^{\circ\circ}$
SVM	**0.8388**$^{\blacktriangle\blacktriangle\blacktriangle}$	**0.7170**$^{\circ\circ\blacktriangle}$	**0.7852**$^{\circ\circ\circ}$	**0.8500**$^{\blacktriangle\circ\blacktriangle}$	**0.8548**$^{\circ\circ\circ}$

sults from the previous section, according to all metrics. The Naive Bayes classifier performs worse than the baseline in terms of P1 and R-precision. SVM clearly outperforms the other algorithms and is able to obtain very high scores; significantly better than the baseline on all metrics. Interestingly, we see that the use of n-gram based reranking has both a precision enhancing effect for J48 and SVM (the P1 and MRR scores go up) and a recall-enhancing effect.

Table 7. Results of removing specific feature types from the training data for the SVM classifier and n-gram based reranking. ▼ and ° indicate that a score is significantly worse or statistically indistinguishable. The leftmost symbol represents the difference with the all features run, the next with the without history features run, and the rightmost symbol the without concept features run.

	P1	R-prec	Recall	MRR	SR
All features	**0.8388**	**0.7170**	**0.7852**	**0.8500**	**0.8548**
Without history	0.7867 $^{\circ}$	0.4687$^{\blacktriangledown}$	0.6272°	0.8009°	0.8041°
Without concept	0.5826$^{\blacktriangledown\circ}$	0.3282$^{\blacktriangledown\circ}$	0.4554$^{\blacktriangledown\circ}$	0.5826$^{\circ\circ}$	0.5826$^{\blacktriangledown\circ}$
Without history and concept	0.1929$^{\blacktriangledown\blacktriangledown\blacktriangledown}$	0.1429$^{\blacktriangledown\blacktriangledown\circ}$	0.1679$^{\blacktriangledown\blacktriangledown\blacktriangledown}$	0.1929$^{\blacktriangledown\blacktriangledown\blacktriangledown}$	0.1929$^{\blacktriangledown\blacktriangledown\blacktriangledown}$

In Section 4.3 we identified several groups of features, relating to the n-gram, concept, their combination, or the session history. We will now zoom in on the performance of these groups. Table 7 shows the results when several of these groups are removed from the training data for the SVM classifier. It turns out that both the n-gram specific and n-gram + concept specific features are needed for classification. When these groups are removed, none of the relevant concepts are identified. From Table 7 we observe that removing the history features results in a drop in performance, albeit a small one. When the concept features are removed, the resulting performance drops even further and their combined removal yields very low scores. Classification without the concept based features results in performance that is statistically indistinguishable from the baseline.

Next we turn to a comparison of n-gram based reranking and full query reranking. Table 8 shows the results when we omit the n-grams and only the full query is used to generate features. We again observe that SVM significantly outperforms J48, NB, as well as the baseline. We further note that these scores are the highest obtained so far and this approach is able to return almost 90% of all relevant concepts. This result is very

Table 8. Results for full query-based reranking. ▲ ▼ and ° indicate that a score is significantly better, worse or statistically indistinguishable respectively. The leftmost symbol represents the difference with the baseline, the next with the J48 run, and the rightmost with the NB run.

Machine learner	P1	R-prec	Recall	MRR	SR
baseline	0.5636	0.5216	0.6768	0.6400	0.7535
J48	0.7055$^{\blacktriangle}$	0.5907°	0.6664°	0.6768°	0.7314°
NB	0.7110$^{\blacktriangle\circ}$	0.6004$^{\circ\circ}$	0.7173$^{\circ\circ}$	0.7121$^{\circ\circ}$	0.7889$^{\circ\circ}$
SVM	**0.8908**$^{\blacktriangle\blacktriangle\blacktriangle}$	**0.8604**$^{\blacktriangle\blacktriangle\blacktriangle}$	**0.8890**$^{\blacktriangle\blacktriangle\blacktriangle}$	**0.8173**$^{\blacktriangle\circ\blacktriangle}$	**0.8963**$^{\blacktriangle\blacktriangle\blacktriangle}$

encouraging and shows that the approach taken handles semantic query suggestions extremely well.

6.3 Feature Selection

Several methods exist with which to automatically determine the most informative features given training instances and their class labels (in our case the class label indicates whether the current concept is selected by the assessors). In this section we report on using the information gain based algorithm for feature selection [38].

Table 9. Results of calculating the information gain with respect to the class label for all features (truncated after 7 features). The higher this score, the more informative a feature is.

N-grams	Full-queries
0.119 $RANK(c, Q)$	0.190 $RANK(c, Q)$
0.107 $DOCID$	0.108 $TEQ(c, Q)$
0.052 $INLINKS(c)$	0.080 $INLINKS(c)$
0.040 $TF_{anchor}(c, Q)$	0.056 $DOCID$
0.038 $OUTLINKS(c)$	0.041 $OUTLINKS(c)$
0.037 $TF_{title}(c, Q)$	0.033 $SCORE(c, Q)$
0.031 $TEQ(c, Q)$	0.025 $REDIRECT(c)$

Table 9 shows the features with the highest information gain scores for both n-gram and full-query based reranking. From this table we observe that the rank at which the retrieval framework puts a concept with respect to an n-gram is most informative. Also the number of in- and outlinks, and whether the n-gram matches the concept's label are informative. $DOCID$ is the internal identifier of each concept and not a feature that we explicitly implemented. However, it turns out that some DBpedia concepts have a higher a priori probability of getting selected. Indeed, in the assessments there were a total of 854 concepts identified, of which 505 are unique. Some of these repetitions are caused because of a coherent information need in the user sessions; when a user rewrites her query by adding or changing part of the query, the remaining concepts remain the same and were annotated as such. As to n-gram based reranking, the term frequency in the title and anchor texts are strong indicators of relevance for given phrase and concept.

7 Conclusion and Future Work

We have introduced the task of semantic query suggestion and presented a method that uses supervised machine learning methods to learn which concepts are used in a query. The concepts are obtained from an ontology and may be used to provide search or navigation suggestions to the user, or as an entry point into the Linked Open Data cloud. Our method extracts query, document, and history specific features from manual annotations and learns how to best rank candidate concepts given an input query.

Our results were obtained using the Dutch version of DBpedia and queries from a log of the Netherlands Institute for Sound and Vision. Although these resources are in Dutch, the framework we have presented is language-independent. Moreover, the approach is also generic in that several of the employed features could be used with ontologies other than DBpedia. However, as became clear from Table 7 and 9, DBpedia related features such as inlinks and redirects were especially helpful. Using Support Vector Machines and features extracted from the full input queries yields optimal results. The best performing run is able to locate almost 90% of the relevant concepts on average. Moreover, this particular run achieves a precision@1 of 89% which means that for this percentage of queries a relevant concept is returned as the first suggestion.[5]

In sum, we have shown that the semantic query suggestion problem can be successfully cast as a ranking problem. The best way of handling query terms is not as separate n-grams, but as a single unit—a finding also interesting from an efficiency viewpoint, since the number of n-grams is quadratic with respect to the length of the query. All types of feature were found to be helpful and, besides document and term features, we found that concept features were also important in achieving our best performance.

As to future work, we will look into expanding the list of features, for example by including more structural features such as ones pertaining to the structure of the ontology. Another question that should be answered is how much training data is needed in order to arrive at a reasonable level of performance. We also intend to go beyond suggesting concepts and look at which part of the query should be linked. Finally, we believe that there might be room for further improvement by using session history in other ways. One option would be a more fine-grained notion of session changes, for example using query overlap [18], or a wider one which considers user history over multiple sessions. Finally, our current approach is trained to find matches between (parts of) the user query and DBpedia concepts, comparable to finding `skos:exactMatch` or even `owl:equivalentClass` relations in an ontology matching task. However, semantic query suggestion can also be interpreted in a broader sense, where not only exact matches but also semantically related concepts are suggested [23]. We believe that our approach can be easily adapted to incorporate such semantically related suggestions.

[5] Our high numbers can be partially explained by the fact that we have decided to focus on the quality of the suggested concepts and as such removed "anomalous" queries from the evaluation, i.e., queries with typos or that were too ambiguous for human assessors to be able to assign a concept to. The influence of this selection on the end-to-end results remains a topic for future work.

Acknowledgments

We thank the anonymous reviewers for their constructive comments. This research was carried out in the context of the Virtual Laboratory for e-Science project and supported by the DuOMAn project carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments under project number STE-09-12 and the Netherlands Organisation for Scientific Research (NWO) under project numbers 017.001.190, 640.001.501, 640.002.501, 612.066.512, 612.061.814, 612.061.815, 640.004.802.

Bibliography

[1] Z. Aleksovski, M. C. A. Klein, W. ten Kate, and F. van Harmelen. Matching unstructured vocabularies using a background ontology. In *EKAW '06*, 2006.

[2] S. Auer and J. Lehmann. What have Innsbruck and Leipzig in common? Extracting semantics from wiki content. In *The Semantic Web: Research and Applications*, 2007.

[3] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. *ISWC '07*, 2007.

[4] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[5] S. M. Beitzel, E. C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder. Hourly analysis of a very large topically categorized web query log. In *SIGIR '04*, 2004.

[6] M. Bendersky and W. B. Croft. Discovering key concepts in verbose queries. In *SIGIR '08*, 2008.

[7] M. Bendersky and W. B. Croft. Analysis of long queries in a large scale search log. In *WSCD '09*, 2009.

[8] A. Bhole, B. Fortuna, M. Grobelnik, and D. Mladenic. Extracting named entities and relating them over time based on wikipedia. *Informatica*, 4(4):463–468, 2007.

[9] C. Bizer, R. Cyganiak, S. Auer, and G. Kobilarov. DBpedia–querying Wikipedia like a database. In *WWW '07*, 2007.

[10] C. Caracciolo, J. Euzenat, L. Hollink, R. Ichise, A. Isaac, V. Malaisé, C. Meilicke, J. Pane, , P. Shvaiko, H. Stuckenschmidt, O. Šváb, and V. Svátek. Results of the OAEI 2008. In *The Third International Workshop on Ontology Matching at ISWC*, 2008.

[11] C. Chemudugunta, A. Holloway, P. Smyth, and M. Steyvers. Modeling documents by combining semantic concepts with unsupervised statistical learning. In *ISWC '08*, 2008.

[12] K. W. Church and W. A. Gale. Inverse document frequency (IDF): A measure of deviations from poisson. In *Proc. Third Workshop on Very Large Corpora*, 1995.

[13] C. Fellbaum, M. Palmer, H. T. Dang, L. Delfs, and S. Wolf. Manual and automatic semantic annotation with wordnet. In *WordNet and Other Lexical Resources*, 2001.

[14] J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *SIGIR '09*, 2009.

[15] D. Hiemstra. *Using Language Models for Information Retrieval*. PhD thesis, University of Twente, 2001.

[16] B. J. Jansen, A. Goodrum, and A. Spink. Searching for multimedia: analysis of audio, video and image web queries. *World Wide Web*, 3(4):249–254, 2000.
[17] B. J. Jansen, A. Spink, and T. Saracevic. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information Processing and Management*, 36(2):207–227, 2000.
[18] B. J. Jansen, A. Spink, C. Blakely, and S. Koshman. Defining a session on web search engines. *J. Am. Soc. Inf. Sci. Technol.*, 58(6):862–871, 2007.
[19] G. H. John and P. Langley. Estimating continuous distributions in bayesian classifiers. In *UAI '95*, 1995.
[20] A. Kiryakov, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff. Semantic annotation, indexing, and retrieval. *J. Web Sem.*, 2(1):49–79, 2004.
[21] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
[22] D. L. Mcguinness. Ontologies come of age. In D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster, editors, *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press, 2003.
[23] E. Meij, P. Mika, and H. Zaragoza. Investigating the demand side of semantic search through query log analysis. In *SemSearch '09*, 2009.
[24] R. Mihalcea and A. Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *CIKM '07*, 2007.
[25] D. Milne and I. H. Witten. Learning to link with wikipedia. In *CIKM '08*, 2008.
[26] G. Mishne and M. de Rijke. A study of blog search. In *ECIR 2006*, 2006.
[27] J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods: support vector learning*. MIT Press, 1999.
[28] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98*, 1998.
[29] R. J. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
[30] P. Shvaiko and J. Euzenat. A survey of schema-based matching approaches. *Journal on Data Semantics*, 4(3730):146–171, 2005.
[31] A. Spink, B. J. Jansen, D. Wolfram, and T. Saracevic. From e-sex to e-commerce: Web search changes. *IEEE Computer*, 35(3):107–109, 2002.
[32] G. Stoilos, G. B. Stamou, and S. D. Kollias. A string metric for ontology alignment. In *ISWC '05*, 2005.
[33] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW '07*, 2007.
[34] W. R. van Hage, M. de Rijke, and M. Marx. Information retrieval support for ontology construction and use. In *ISWC '04*, 2004.
[35] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, 1995.
[36] S. Wang, G. Englebienne, and S. Schlobach. Learning concept mappings from instance similarity. In *ISWC '08*, 2008.
[37] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.
[38] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML '97*, 1997.

[39] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.

[40] Y. Zhou and B. W. Croft. Query performance prediction in web search environments. In *SIGIR '07*, 2007.