Web Semantics: Science, Services and Agents on the World Wide Web 9 (2011) 418–433

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: http://www.elsevier.com/locate/websem

Mapping queries to the Linking Open Data cloud: A case study using DBpedia ☆

Edgar Meij [a,*], Marc Bron [a], Laura Hollink [b], Bouke Huurnink [a], Maarten de Rijke [a]

[a] ISLA, University of Amsterdam, Science Park 107, Amsterdam, The Netherlands
[b] Web Information Systems, Delft University of Technology, Mekelweg 4, Delft, The Netherlands

A R T I C L E I N F O

Article history:
Available online 28 April 2011

Keywords:
Semantic query analysis
Linking Open Data
Machine learning
Information retrieval

A B S T R A C T

We introduce the task of mapping search engine queries to DBpedia, a major linking hub in the Linking Open Data cloud. We propose and compare various methods for addressing this task, using a mixture of information retrieval and machine learning techniques. Specifically, we present a supervised machine learning-based method to determine which concepts are intended by a user issuing a query. The concepts are obtained from an ontology and may be used to provide contextual information, related concepts, or navigational suggestions to the user submitting the query. Our approach first ranks candidate concepts using a language modeling for information retrieval framework. We then extract query, concept, and search-history feature vectors for these concepts. Using manual annotations we inform a machine learning algorithm that learns how to select concepts from the candidates given an input query. Simply performing a lexical match between the queries and concepts is found to perform poorly and so does using retrieval alone, i.e., omitting the concept selection stage. Our proposed method significantly improves upon these baselines and we find that support vector machines are able to achieve the best performance out of the machine learning algorithms evaluated.

1. Introduction

A significant task in building and maintaining the Semantic Web is link generation. Links allow a person or machine to explore and understand the web of data more easily: when you have linked data, you can find related data [2]. The Linking Open Data (LOD) [2–4] initiative extends the web by publishing various open data sets and by setting links between items (or concepts) from different data sources in a (semi) automated fashion [5–7]. The resulting data commons is termed the Linking Open Data cloud, and provides a key ingredient for realizing the Semantic Web. By now, the LOD cloud contains millions of concepts from over one hundred structured data sets.

Unstructured data resources—such as textual documents or queries submitted to a search engine—can be enriched by mapping their content to structured knowledge repositories like the LOD cloud. This type of enrichment may serve multiple goals, such as explicit anchoring of the data resources in background knowledge or ontology learning and population. The former enables new forms of intelligent search and browsing; authors or readers of a piece of text may find mappings to the LOD cloud to supply useful pointers, for example, to concepts capturing or relating to the contents of the document. In ontology learning applications, mappings may be used to learn new concepts or relations between them [8]. Recently, data-driven methods have been proposed to map phrases appearing in full-text documents to Wikipedia articles. For example, Mihalcea and Csomai [9] propose incorporating linguistic features in a machine learning framework to map phrases in full-text documents to Wikipedia articles—this approach is further improved upon by Milne and Witten [10]. Because of the connection between Wikipedia and DBpedia [6], such data-driven methods help us to establish links between textual documents and the LOD cloud, with DBpedia being one of the key interlinking hubs. Indeed, we consider DBpedia to be a major linking hub of the LOD cloud and, as such, a perfect entry point.

Search engine queries are one type of unstructured data that could benefit from being mapped to a structured knowledge base such as DBpedia. Semantic mappings of this kind can be used to support users in their search and browsing activities, for example by (i) helping the user acquire contextual information, (ii) suggesting related concepts or associated terms that may be used for search, and (iii) providing valuable navigational suggestions. In the context of web search, various methods exist for helping the user formulate his or her queries [11–13]. For example, the Yahoo! search interface features a so-called "searchassist," that suggests important phrases in response to a query. These suggestions lack any semantics, however, which we address in this paper by mapping queries to DBpedia concepts. In the case of a specialized

☆ This paper is an extended and revised version of [1].
* Corresponding author. Tel.: +31 205257565; fax: +31 205257490.
E-mail addresses: edgar.meij@uva.nl (E. Meij), m.m.bron@uva.nl (M. Bron), l.hollink@tudelft.nl (L. Hollink), bhuurnink@uva.nl (B. Huurnink), derijke@uva.nl (M. de Rijke).

1570-8268/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.websem.2011.04.001

search engine with accompanying knowledge base, automatic mappings between natural language queries and concepts aid the user in exploring the contents of both the collection and the knowledge base [14]. They can also help a novice user understand the structure and specific nomenclature of the domain. Furthermore, when the items to be retrieved are also annotated (e.g., using concepts from the LOD cloud through RDFa, microformats, or any other kind of annotation framework), the semantic mappings on the queries can be used to facilitate matching at the semantic level or an advanced form of query-based faceted result presentation. This can partly be achieved by simply using a richer indexing strategy of the items in the collection together with conventional querying mechanisms. Generating conceptual mappings for the queries, however, can improve the matching and help clarify the structure of the domain to the end user.

Once a mapping has been established, the links between a query and a knowledge repository can be used to create semantic profiles of users based on the queries they issue. They can also be exploited to enrich items in the LOD cloud, for instance by viewing a query as a (user-generated) annotation of the items it has been linked to, similar to the way in which a query can be used to label images that a user clicks on as the result of a search [15]. This type of annotation can, for example, be used to discover aspects or facets of concepts [16]. In this paper, we focus on the task of automatically mapping free text search engine queries to the LOD cloud, in particular DBpedia. As an example of the task, consider the query "obama white house." The query mapping algorithm we envision should return links to the concepts labeled BARACK OBAMA and WHITE HOUSE.

Queries submitted to a search engine are particularly challenging to map to structured knowledge sources, as they are much shorter than typical documents and tend to consist of only a few terms [11,17]. Their length implies that we have far less context than in regular text documents. Hence, we cannot use previously established approaches such as shallow parsing or part-of-speech tagging [9]. To address these issues, we propose a novel method that leverages the textual representation of each concept as well as query-based and concept-based features in a machine learning framework. On the other hand, working with search engine queries entails that we do have search history information available that may provide contextual anchoring. In this paper, we employ this query-specific kind of context as a separate feature type.

Our approach can be summarized as follows. First, given a query, we use language modeling for information retrieval (IR) to retrieve the most relevant concepts as potential targets for mapping. We then use supervised machine learning methods to decide which of the retrieved concepts should be mapped and which should be discarded. In order to train the machine learner, we examined close to 1000 search engine queries and manually mapped over 600 of these to relevant concepts in DBpedia.[1]

The research questions we address are the following.

1. Can we successfully address the task of mapping search engine queries to ontological concepts using a combination of information retrieval and machine learning techniques? *A typical approach for mapping text to concepts is to apply some form of lexical matching between concept labels and terms. What are the results of applying this method to our task? What are the results when using a purely retrieval-based approach? How do these results compare to those of our proposed method?*
2. What is the best way of handling the input query; what are the effects on performance when we map parts of the query instead of the query in its entirety?
3. As input to the machine learning algorithms we extract and compute a wide variety of features, pertaining to the query terms, concepts, and search history. Which feature type helps most? Which individual feature is most informative?
4. Machine learning generally comes with a number of parameter settings. We ask: what are the effects of varying these parameters? *What are the effects when varying the size of the training set, the fraction of positive examples, as well as any algorithm-specific parameters? Furthermore, we provide the machine learning step with a small set of candidate concepts. What are the effects of varying the size of this set?*

Our main contributions are as follows. We propose and evaluate two variations of a novel and effective approach for mapping queries to DBpedia and, hence, the LOD cloud. We accompany this with an extensive analysis of the results, of the robustness of our methods, and of the contributions of the features used. We also facilitate future work on the problem by making our used resources publicly available.

The remainder of this paper is structured as follows. In Section 2 we discuss related work. Sections 3 and 4 detail the query mapping task and our approach. Our experimental setup is described in Section 5 and our results are presented in Section 6. Section 7 follows with a discussion and detailed analysis of the results and we end with a concluding section.

2. Related work

Mapping terms or phrases to ontologies is related to several areas of research. These include Semantic Web areas such as ontology learning, population, and matching and semantic annotation, but also areas from language technology, information retrieval, and natural language interfaces to databases.

2.1. Natural language interfaces to databases

The first body of related work that we discuss is from the field of natural language interfaces to databases [18]. For example, BANKS [19], DISCOVER [20], and DBXplorer [21] allow novice users to query large, complex databases using a set of keywords. Tata and Lohman [22] propose a similar keyword-based querying mechanism but with additional aggregation facilities. All of these systems perform some kind of matching between all keywords in the input query and the contents of the database. If any matches are found, they are joined together to form tuple trees. A tuple tree contains all the keywords and is considered a potential answer. Note that when all query keywords appear together in a single record, there is no need for any joins. The result in these systems is generally a list of such tuple trees, much like a search engine. The actual matching function varies per system but boils down to determining literal matches between each keyword and the columns/rows of each table. This is exactly the approach taken by our first baseline. Our second baseline uses IR techniques to improve upon this form of lexical matching. Our proposed method does not perform any joins in its current form but, in contrast to ours, none of these earlier systems apply any kind of term weighing or machine learning.

NAGA is a similar system that is more tied to the Semantic Web [23,24]. It uses language modeling intuitions to determine a ranking of possible answer graphs, based on the frequency of occurrence of terms in the knowledge base. This scoring mechanism has been shown to perform better than that of BANKS on various

[1] The queries, human assessments, and extracted features are publicly available for download at URL `http://ilps.science.uva.nl/resources/jws10_annotations`.

test collections [23]. NAGA does not support approximate matching and keyword-augmented queries. Our method, on the other hand, takes as input any unstructured keyword query.

Demidova et al. [25] present the evaluation of a system that maps keyword queries to structured query templates. The query terms are mapped to specific places in each template and the templates are subsequently ranked, explicitly taking diversity into account. They find that applying diversification to query template ranking achieves a significant reduction of result redundancy. Kaufmann and Bernstein [26] perform a user study in which they evaluate various natural language interfaces to structured knowledge bases. Each interface has a different level of complexity and the task they ask their users to accomplish is to rewrite a set of factoid and list queries for each interface, with the goal of answering each question using the contents of the knowledge base. They find that for this task, the optimal strategy is a combination of structure (in the form of a fixed set of question beginnings, such as "How many ... " and "Which ... ") and free text. Our task is more general than the task evaluated in Kaufmann and Bernstein [26], in that we do not investigate if, how well, or how easily the users' queries are answered, but whether they are mapped to the right concepts. We postulate various benefits of these mappings other than to answering questions, such as to provide contextual suggestions, to start exploring the knowledge base, etcetera.

2.2. Ontology matching

In *ontology matching*, relations between concepts from different ontologies are identified. The Ontology Alignment Evaluation Initiative has addressed this task since 2008. Here, participants link a largely unstructured thesaurus to DBpedia [27]. The relations to be obtained 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 WordNet or an upper ontology [28]. E.g., Wang et al. [29] 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 use neither semantic structure nor instances (e.g., [30]). Aleksovski et al. [31] use a lexical comparison of labels to map both the source and the target ontology to a semantically rich external source of background knowledge. This type of matching is referred to as "lexical matching" and is used in cases where the ontologies do not have any instances or structure. Lexical matching is very similar to our task, as we do not have any semantic structure in the queries. Indeed, the queries that we link are free text utterances (submitted as queries to a search engine) instead of standardized concept labels, which makes our task intrinsically harder. In order to validate our method, we use lexical matching as one of the baselines to which we compare our approach.

2.3. Ontology learning, ontology population, and semantic annotation

In the field of *ontology learning and population*, concepts and/or their instances are learned from unstructured or semi-structured documents, together with links between concepts [32]. Well-known examples of ontology learning tools are OntoGen [33] and TextToOnto [34]. More related to our task is the work done on *semantic annotation*, the process of mapping text from unstructured data resources to concepts from ontologies or other sources of structured knowledge. In the simplest case, this is performed using a lexical match between the labels of each candidate concept and the contents of the text [13,35–37]. A well-known example of a more elaborate approach is Ontotext's KIM platform [38]. The KIM platform builds on GATE to detect named entities and to link them to concepts in an ontology [39]. Entities unknown to the ontology are given a URL and are fed back into the ontology, thus populating it further. OpenCalais[2] provides semantic annotations of textual documents by automatically identifying entities, events, and facts. Each annotation is given a URI that is linked to concepts from the LOD cloud when possible. Bhole et al. [40] describe another example of semantic document analysis, where named entities are related over time using Wikipedia. Chemudugunta et al. [41] do not restrict themselves to named entities, but instead use topic models to link all words in a document to ontological concepts. Other sub-problems of semantic annotation include sense tagging and word sense disambiguation [42]. Some of the techniques developed there have fed into automatic link generation between full-text documents and Wikipedia. For example, Milne and Witten [10], building on the work of Mihalcea and Csomai [9], depend heavily on contextual information from terms and phrases surrounding the source text to determine the best Wikipedia articles to link to. The authors apply part-of-speech tagging and develop several ranking procedures for candidate Wikipedia articles. Our approach differs from these approaches in that we do not limit ourselves to exact matches with the query terms (although that method is one of our baselines). Another distinct difference is that we utilize much sparser data in the form of user queries, as opposed to full-text documents. Hence, we cannot 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 instead uses search session history to obtain contextual information.

2.4. Semantic query analysis

Turning to *semantic query analysis* (as opposed to semantic analysis of full documents), Guo et al. [43] perform named entity recognition in queries; they recognize a single entity in each query and subsequently classify it into one of a very 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 concepts in DBpedia. Several other approaches have been proposed that link queries to a small set of categories. Mishne and de Rijke [44] use online product search engines to link queries to product categories; Beitzel et al. [45] link millions of queries to 17 topical categories based on a list of manually pre-categorized queries; Jansen et al. [46] use commonly occurring multimedia terms to categorize audio, video, and image queries; and Huurnink et al. [47] utilize structured data from clicked results to link queries in a multimedia archive to an in-house thesaurus.

Many applications of (semantic) query analysis have been proposed, such as disambiguation [48,49] and rewriting. Jansen et al. [11] use query logs to determine which queries or query rewrites occur frequently. Others perform query analysis and try to identify the most relevant terms [50], to predict the query's performance a priori [51], or combine the two [52]. Bendersky and Croft [50] 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 that convey the most information in a query and contribute most to the resulting retrieval performance. Our approach differs in that we link queries to a structured knowledge base instead. We incorporate and evaluate several of the features proposed in [50–52] on our task below.

3. The task

The query mapping task that we address in this paper is the following. Given a query submitted to a search engine, identify the

[2] http://www.opencalais.com/.

underlying concepts that are intended or implied by the user issuing the query, where the concepts are taken from a structured knowledge base. We address our task in the setting of a digital archive, specifically, 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 updated daily with new television and radio broadcasts. Users of the archive's search facilities consist primarily of media professionals who use the online search interface to locate audiovisual items to be used in new programs such as documentaries and news reviews. The contents of the audiovisual items are diverse and cover a wide range of topics, people, places, and more. Furthermore, a significant part (around 50%) of the query terms are informational consisting of either general keywords (typically noun phrases such as "war," "soccer," "forest fire," and "children") or proper names [47].

Because of its central role in the Linking 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 semantically related to the query. These concepts can then be used, for example, to suggest relevant multimedia items associated with each concept, to suggest linked geodata from the LOD cloud, or to suggest contextual information, such as text snippets from a Wikipedia article.

4. Approach

Our approach for mapping search engine queries to concepts consists of two stages. In the first stage, we select a set of candidate concepts. In the second stage, we use supervised machine learning to classify each candidate concept as being intended by the query or not.

In order to find candidate concepts in the first stage, we leverage the textual descriptions (`rdfs:comment` and/or `dbpprop:abstract` in the case of DBpedia) of the concepts as each description of a concept may contain related words, synonyms, or alternative terms that refer to the concept. An example is given in Table 1. From this example it is clear that the use of such properties for retrieval improves recall (we find BARACK OBAMA using the terms "President of the United States") at the cost of precision (we also find BARACK OBAMA when searching for "John McCain"). In order to use the concept descriptions, we adopt a language modeling for information retrieval framework to create a ranked list of candidate concepts. This framework will be further introduced in Section 4.1.

Since we are dealing with an ontology extracted from Wikipedia, we have several options with respect to which textual representation(s) we use. The possibilities include: (i) the title of the article (similar to a lexical matching approach where only the `rdfs:label` is used), (ii) the first sentence or paragraph of an article (where a definition should be provided according to the Wikipedia guidelines [53]), (iii) the full text of the article, (iv) the anchor texts of the incoming hyperlinks from other articles, and (v) a combination of any of these. For our experiments we aim to maximize recall and use the combination of all available fields with or without the incoming anchor texts. In Section 7.2 we discuss the relative performance of each field and of their combinations.

For the first stage, we also vary the way we handle the query. In the simplest case, we take the query as is and retrieve concepts for the query in its entirety. As an alternative, we consider extracting all possible n-grams from the query, generating a ranked list for each, and merging the results. An example of what happens when we vary the query representation is given in Table 2 for the query "obama white house." From this example it is clear why we differentiate between the two ways of representing the query. If we simply use the full query on its own (first row), we 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
Example DBpedia representation of the concept BARACK OBAMA.

Property	Value
`rdfs:comment`	Barack Hussein Obama II (born August 4, 1961) is the 44th and current President of the United States. The first African American to hold the office, he previously served as the junior United States Senator from Illinois from January 2005 until he resigned after his election to the presidency in November 2008. Obama is a graduate of Columbia University and Harvard Law School, where he was the president of the Harvard Law Review.
`dbpprop:abstract`	Barack Hussein Obama II (born August 4, 1961) is the 44th and current President of the United States. The first African American to hold the office, he previously served as the junior United States Senator from Illinois from January 2005 until he resigned after his election to the presidency in November 2008. Obama is a graduate of Columbia University and Harvard Law School, where he was the president of the Harvard Law Review. He was a community organizer in Chicago before earning his law degree. He worked as a civil rights attorney in Chicago and taught constitutional law at the University of Chicago Law School from 1992 to 2004. Obama served three terms in the Illinois Senate from 1997 to 2004. Following an unsuccessful bid for a seat in the U.S. House of Representatives in 2000, Obama ran for United States Senate in 2004. His victory, from a crowded field, in the March 2004 Democratic primary raised his visibility. His prime-time televised keynote address at the Democratic National Convention in July 2004 made him a rising star nationally in the Democratic Party. He was elected to the U.S. Senate in November 2004 by the largest margin in the history of Illinois. He began his run for the presidency in February 2007. After a close campaign in the 2008 Democratic Party presidential primaries against Hillary Rodham Clinton, he won his party's nomination, becoming the first major party African American candidate for president. In the 2008 general election, he defeated Republican nominee John McCain and was inaugurated as president on January 20, 2009.

In the second stage, a supervised machine learning approach is used to classify each candidate concept as either relevant or non-relevant or, in other words, to decide which of the candidate concepts from the first stage should be kept as viable concepts for the query in question. In order to create training material for the machine learning algorithms, we asked human annotators to assess search engine queries and manually map them to relevant DBpedia concepts. More details about the test collection and manual annotations are provided in Section 5. The machine learning algorithms we consider are Naive Bayes, Decision Trees, and Support Vector Machines [54,55] which are further detailed in Section 4.2. As input for the machine learning algorithms we need to extract a number of features. We consider features pertaining to the query, concept, their combination, and the session in which the query appears; these are specified 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 highly competitive in terms of retrieval effectiveness [56–58]. Here, a query is viewed as having been generated from a multinomial language model underlying the document, where

Table 2
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 WHITE; PRESIDENT COOLIDGE
obama	**BARACK OBAMA**; MICHELLE OBAMA; PRESIDENTIAL ELECTIONS 2008; HILLARY CLINTON
white	COLONEL WHITE; EDWARD WHITE; WHITECOUNTY; WHITE PLAINS ROAD LINE
house	HOUSE; ROYAL OPERA HOUSE; SYDNEY OPERA HOUSE; FULL HOUSE

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 document language model. These word probabilities are estimated from the document itself (using maximum likelihood estimation) and combined with background collection statistics to overcome zero probability and data sparsity issues; a process known as *smoothing*.

For the n-gram based scoring method, we extract all n-grams from each query $\mathbf{Q}$ (where $1 \leqslant n \leqslant |\mathbf{Q}|$) 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 Q can then be formulated as follows. Each concept c should be ranked according to the probability $P(c|Q)$ that it was generated by the n-gram, which can be rewritten using Bayes' rule as:

$$P(c|Q) = \frac{P(Q|c)P(c)}{P(Q)}. \tag{1}$$

Here, for a fixed n-gram Q, the term $P(Q)$ is the same for all concepts and can 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 information retrieval [59], we obtain

$$P(c|Q) \propto P(c) \prod_{q \in Q} P(q|c)^{n(q,Q)}, \tag{2}$$

where $n(q, Q)$ indicates the count of term q in Q. The probability $P(q|c)$ is smoothed using Bayes smoothing with a Dirichlet prior [58], which is formulated as:

$$P(q|c) = \frac{n(q,c) + \mu P(q)}{\mu + \sum_{q'} n(q',c)}, \tag{3}$$

where $P(q)$ indicates the probability of observing q in a large background collection; $n(q,c)$ is the count of term q in the textual representation of c; μ is a hyperparameter that controls the influence of the background corpus.

4.2. Learning to select concepts

Once we have obtained a ranked list of possible concepts for each n-gram, 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, that takes as input a set of labeled examples (query to concept mappings) and several features of these examples (detailed below). More formally, each query $\mathbf{Q}$ is associated with a ranked list of concepts $\mathbf{c}$ and a set of associated relevance assessments for the concepts. The latter is created by considering all concepts that any annotator used to map $\mathbf{Q}$ to c. If a concept was not selected by any of the annotators, we consider it to be non-relevant for $\mathbf{Q}$. Then, for each query in the set of annotated queries, we consider each combination of n-gram Q and concept c an instance for which we create a feature vector.

The goal of the machine learning algorithm is to learn a function that outputs a relevance status for any new n-gram and concept pair given a feature vector of this new instance. 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 particular machine learning algorithm.

4.3. Features used

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

4.3.1. N-gram features

These features are based on information from an n-gram and are listed in Table 3 (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 total number of concepts and $df(Q)$ the number of concepts in which Q occurs [59]. $WIG(Q)$ indicates the weighted information gain, that was proposed by Zhou and Croft [51] as a predictor of the retrieval performance of a query. It 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 the n-gram Q appears in the entire query logs as a complete or partial query, respectively.

4.3.2. Concept features

Table 3 (second group) lists the features related to a DBpedia concept. This set of features is related to the knowledge we have of the 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`).

4.3.3. N-gram + concept features

This set of features considers the combination of an n-gram and a concept (Table 3, third group). We consider the relative frequency of occurrence of the n-gram as a phrase in the Wikipedia article corresponding to the concept, 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 [59]. Of these, *RIDF* [60] is the difference between expected and observed IDF for a concept, which is defined as

$$RIDF(c,Q) = \log\left(\frac{|Coll|}{df(Q)}\right) + \log\left(1 - \exp\left(\frac{-n(Q,Coll)}{|Coll|}\right)\right).$$

We also consider whether the label of the concept (`rdfs:label`) matches Q in any way and we include the retrieval score and rank as determined by using Eq. (2).

Table 3
Features used, grouped by type. More detailed descriptions in Section 4.3.

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 the query log
$QP(Q)$	Number of times Q appeared as *partial* query in the 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 the SKOS category hierarchy [10]
$CAT(c)$	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 representationfof 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 the label of c contain Q?
$TEQ(c, Q)$	Does the 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

Table 4
An example of queries issued in a (partial) session, translated to English.

Session ID	Query ID	Query (**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

4.3.4. History features

Finally, we consider features based on the previous queries that were issued in the same session (Table 3, fourth group). These features indicate whether the current candidate concept or n-gram occur (partially) in the previously issued queries or retrieved candidate concepts, respectively.

In Section 6 we compare the effectiveness of the feature types listed above for our task, while in Section 7.5 we discuss the relative importance of each individual feature.

5. Experimental setup

In this section we introduce the experimental environment and the experiments that we perform to answer the research questions listed in Section 1. We start with detailing our data sets and then introduce our evaluation measures and manual assessments. We use the Lemur Toolkit for all our language modeling calculations, which efficiently handles very large text collections [61].[3]

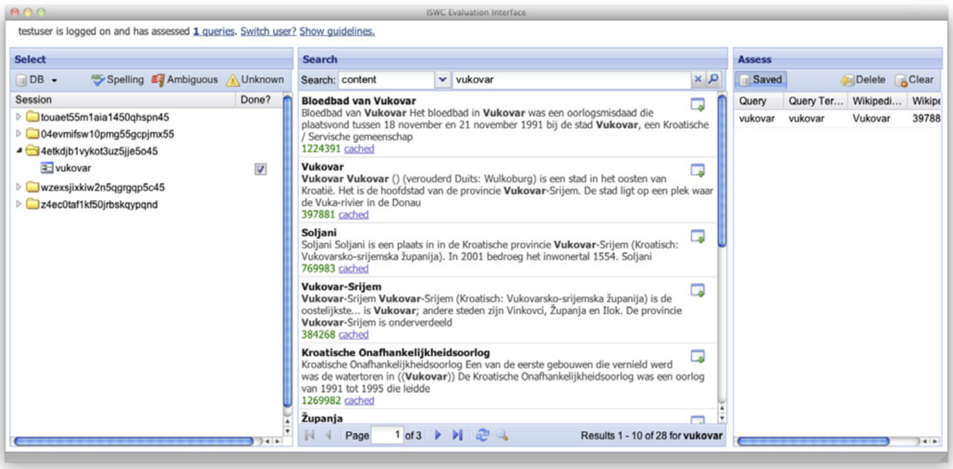
5.1. Data

Two main types of data are needed for our experiments, namely search engine queries and a structured knowledge repository. We have access to a set of 264,503 queries issued between 18 November 2008 to 15 May 2009 to the audiovisual catalog maintained by Sound and Vision. Sound and Vision logs the actions of users on the site, generating session identifiers and time stamps. This allows for 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 4. All queries are Dutch language queries (although we emphasize that nothing in our approach is language dependent). As the "history" of a query, we take all queries previously issued in the same user session. The DBpedia version we use is the most recently issued Dutch language release (3.2). We also 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, five assessors were asked to manually map queries to DBpedia concepts using the interface depicted in Fig. 1. 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 (in the context of the session) for each query therein. 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 at all. For our experiments, we removed all the assessed queries in these "anomalous" categories and were left with a total of 629 assessed queries (out of 998 in total) in 193 randomly selected sessions. In our experiments we primarily focus on evaluating the actual mappings to DBpedia and discard queries which the assessors deemed too anomalous to confidently map to any concept. In this subset, the average query length is 2.14 terms per query and each query has 1.34 concepts annotated on average. In Section 7.1 we report on the inter-annotator agreement.

[3] See http://sourceforge.net/projects/lemur/.



testuser is logged on and has assessed 1 queries. Switch user? Show guidelines.

Session	Done?
touaet55m1aia1450qhspn45	
04evmifsw10pmg55gcpjmx55	
4etkdjb1vykot3uz5jje5o45	
vukovar	✓
wzexsjixkiw2n5qgrgqp5c45	
z4ec0taf1kf50jrbskqypqnd	

Search: content vukovar

Bloedbad van Vukovar
Bloedbad van **Vukovar** Het bloedbad in **Vukovar** was een oorlogsmisdaad die plaatsvond tussen 18 november en 21 november 1991 bij de stad **Vukovar**, een Kroatische / Servische gemeenschap
1224391 cached

Vukovar
Vukovar Vukovar () (verouderd Duits: Wulkoburg) is een stad in het oosten van Kroatië. Het is de hoofdstad van de provincie **Vukovar**-Srijem. De stad ligt op een plek waar de Vuka-rivier in de Donau
397881 cached

Soljani
Soljani Soljani is een plaats in in de Kroatische provincie **Vukovar**-Srijem (Kroatisch: Vukovarsko-srijemska županija). In 2001 bedroeg het inwonertal 1554. Soljani
769983 cached

Vukovar-Srijem
Vukovar-Srijem **Vukovar**-Srijem (Kroatisch: Vukovarsko-srijemska županija) is de oostelijkste... is **Vukovar**; andere steden zijn Vinkovci, Županja en Ilok. De provincie **Vukovar**-Srijem is onderverdeeld
384268 cached

Kroatische Onafhankelijkheidsoorlog
Kroatische Onafhankelijkheidsoorlog Een van de eerste gebouwen die vernield werd was de watertoren in ((**Vukovar**)) De Kroatische Onafhankelijkheidsoorlog was een oorlog van 1991 tot 1995 die leidde
1269982 cached

Županja

Page 1 of 3 Results 1 - 10 of 28 for **vukovar**

Query	Query Ter...	Wikipedi...	Wikipe
vukovar	vukovar	Vukovar	39788

Fig. 1. Screen dump of the web interface the annotators used to manually link queries to concepts. On the left the sessions, in the middle a full-text retrieval interface, and on the right the made annotations.

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 [58], i.e., $\mu = 315$ (cf. Eq. (3)). Initially, we select the 5 highest ranked concepts as input for the concept selection stage. In Section 7.3.1 we report on the influence of varying the number of highest ranked concepts used as input.

As indicated earlier in Section 4.2, we use the following three supervised machine learning algorithms for the concept selection stage: J48, Naive Bayes and Support Vector Machines. The implementations are taken from the Weka machine learning toolkit [54]. J48 is a decision tree algorithm and the Weka implementation of C4.5 [62]. The Naive Bayes classifier uses the training data to estimate the probability that an instance belongs to the target class, given the presence 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 [63]. SVM uses a sequential minimal optimization algorithm to minimize the distance between the hyperplanes which best separate the instances belonging to different classes, as described in [64]. In the experiments in the next section we use a linear kernel. In Section 7.3.4 we discuss the influence of different parameter settings to see whether fine-grained parameter tuning of the algorithms has any significant impact on the end results.

5.4. Testing and evaluation

We define the mapping of search engine queries to DBpedia as a ranking problem. The system that implements a solution to this problem has to return a ranked list of concepts for a given input query, where a higher rank indicates a higher degree of relevance of the concept to the query. The best performing method puts the most relevant concepts towards the top of the ranking. The assessments described above are used to determine the relevance status of each of the concepts with respect to a query.

We employ several measures that are well-known in the field of information retrieval [59], 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 (the percentage of relevant concepts that 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 generalizability of our approach, we perform 10-fold cross validation [54]. This also reduces the possibility of errors being caused by artifacts in the data. Thus, we use 90% of the annotated queries for training and validation and the remainder for testing in each of the folds. The reported scores are averaged over all folds, and all evaluation measures are averaged over the queries used for testing. In Section 7.3.3 we discuss what happens when we vary the size of the folds.

For determining the statistical significance of the observed differences between runs we use one-way ANOVA to determine if there is a significant difference ($\alpha \leqslant 0.05$). We then use the Tukey–Kramer test to determine which of the individual pairs are significantly different. We indicate the best result in each table of results in bold face.

6. Results

In the remainder of this section we report on the experimental results and use them to answer the research questions from Section 1. Here, we compare the following approaches for mapping queries to DBpedia:

(i) a baseline that retrieves only those concepts whose label *lexically matches* the query,
(ii) a *retrieval baseline* that retrieves concepts based solely on their textual representation in the form of the associated Wikipedia article with varying textual fields,
(iii) *n-gram based reranking* that extracts all n-grams from the query and uses machine learning to identify the best concepts, and
(iv) *full query based reranking* that does not extract n-grams, but calculates feature vectors based on the full query and uses machine learning to identify the best concepts.

In the next section we further analyze the results along multiple dimensions, including the effects of varying the number of retrieved concepts in the first stage, varying parameters in the machine learning models, the most informative individual features and types, and the kind of errors that are made by the machine learning algorithms.

6.1. Lexical match

As our first baseline we consider a simple heuristic which is commonly used [35–37] and select concepts that lexically match the query, subject to various constraints. This returns concepts where consecutive terms in the `rdfs:label` are contained in the query or vice versa. An example for the query "joseph haydn" is given in Table 5. We then rank the concepts based on the language modeling score of their associated Wikipedia article given the query (cf. Eq. (2)).

Table 6 shows the scores when using lexical matching for mapping search engine queries. The results in the first row are obtained by only considering the concepts whose label is contained in the query (QCL). This is a frequently taken but naive approach and does not perform well, achieving a P1 score of under 40%. The second row relaxes this constraint and also selects concepts where the query is contained in the concept label (QCL-LCQ). This improves the performance somewhat.

One issue these approaches might have, however, is that they might match parts of compound terms. For example, the query "brooklyn bridge" might not only match the concept BROOKLYN BRIDGE but also the concepts BROOKLYN and BRIDGE. The approach taken for the third row (QCL-LSO) therefore extracts all n-grams from the query, sorts them by the number of terms, and checks whether the label is contained in each of them. If a match is found, the remaining, smaller n-grams are skipped.

The last row ("oracle") shows the results when we initially select all concepts whose terms in the label matches with any part of the query. Then, we keep only those concepts that were annotated by the assessors. As such, it indicates the upper bound on the performance that lexical matching might obtain. From the low absolute scores we conclude that, although lexical matching is a common approach for matching unstructured text with structured data, it does not perform well for our task and we need to consider additional kinds of information pertaining to each concept.

6.2. Retrieval only

As our second baseline, we take the entire query as issued by the user and employ Eq. (2) to rank DBpedia concepts based on their textual representation; this technique is similar to using a search engine and performing a search within Wikipedia. We use either the textual contents of the Wikipedia article ("content-only"—which includes only the article's text) or a combination of the article's text, the title, and the anchor texts of incoming links ("full text").

Table 7 shows the results of this method. We note that including the title and anchor texts of the incoming links results in improved retrieval performance overall. This is a strong baseline; on average, over 65% of the relevant concepts are correctly identified in the top-5 and, furthermore, 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. In Section 7.2 we further discuss the relative performance of each textual representation as well as various combinations.

6.3. N-gram based concept selection

Table 8 (last row) shows the concepts obtained for the second baseline and the query "challenger wubbo ockels." Here, two relevant concepts are retrieved at ranks 1 and 4. When we look at the same results for all possible 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 given earlier in Table 2 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 machine learning algorithms to decide which of the suggested concepts to keep. The features used here are described in Section 4.2.

Table 9 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 results 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 scores that are very high, 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.

6.4. Full query-based concept selection

Next, we turn to a comparison of n-gram based and full-query based concept selection. Using the full-query based concept selec-

Table 5
An example of the concepts obtained using lexical matching for the query "joseph haydn."

QCL	QCL-LCQ	QCL-LSO
JOSEPH HAYDN	JOSEPH HAYDN	JOSEPH HAYDN
JOSEPH	JOSEPH	
	JOSEPH HAYDN OPERAS	
	JOSEPH HAYDN SYMPHONIES	

Table 6
Lexical match baseline results using lexical matching between labels and query to select concepts.

	P1	R-prec	Recall	MRR	SR
QCL	0.3956	0.3140	0.4282	0.4117	0.4882
QCL-LCQ	0.4286	0.3485	0.4881	0.4564	0.5479
QCL-LSO	0.4160	0.2747	0.3435	0.3775	0.4160
Oracle	**0.5808**	**0.4560**	**0.5902**	**0.5380**	**0.6672**

Table 7
Retrieval only baseline results which ranks concepts using the entire query **Q** and either the content of the Wikipedia article or the full text associated with each DBpedia concept (including title and anchor texts of incoming hyperlinks).

	P1	R-prec	Recall	MRR	SR
Full text	**0.5636**	**0.5216**	**0.6768**	**0.6400**	**0.7535**
Content-only	0.5510	0.5134	0.6632	0.6252	0.7363

Table 8
An example of the concepts obtained when using retrieval only for the n-grams in the query "challenger wubbo ockels," ranked by retrieval score. Concepts annotated by the human annotators for this query in boldface.

N-gram	Candidate concepts
challenger	**SPACE SHUTTLE CHALLENGER**; CHALLENGER; BOMBARDIER CHALLENGER; STS-61-A; STS-9
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; ANDRÉ KUIPERS
challenger wubbo ockels	**WUBBO OCKELS**; STS-61-A; SPACELAB; **SPACE SHUTTLE CHALLENGER**; STS-9

Table 9
Results for n-gram based concept selection. ▲, ▼ 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.

	P1	R-prec	Recall	MRR	SR
Baseline	0.5636	0.5216	0.6768	0.6400	0.7535
J48	0.6586°	0.5648°	0.7253°	$0.7348^{\blacktriangle}$	0.7989°
NB	$0.4494^{\blacktriangledown\blacktriangledown}$	$0.4088^{\blacktriangledown\blacktriangledown}$	$0.6948^{\circ\circ}$	$0.7278^{\circ\circ}$	$0.7710^{\circ\circ}$
SVM	$\mathbf{0.7998}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.6718}^{\blacktriangle\circ\blacktriangle}$	$\mathbf{0.7556}^{\circ\circ\circ}$	$\mathbf{0.8131}^{\blacktriangle\circ\circ}$	$\mathbf{0.8240}^{\circ\circ\circ}$

Table 10
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.

	P1	R-prec	Recall	MRR	SR
Baseline	0.5636	0.5216	0.6768	0.6400	0.7535
J48	$0.7152^{\blacktriangle}$	5857°	0.6597°	0.6877°	0.7317°
NB	$0.6925^{\blacktriangle\circ}$	$0.5897^{\circ\circ}$	$0.6865^{\circ\circ}$	$0.6989^{\circ\circ}$	$0.7626^{\circ\circ}$
SVM	$\mathbf{0.8833}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.8666}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.8975}^{\blacktriangle\blacktriangle\blacktriangle}$	$\mathbf{0.8406}^{\blacktriangle\circ\blacktriangle}$	$\mathbf{0.9053}^{\blacktriangle\blacktriangle\blacktriangle}$

tion method, we take each query as is (an example is given in the last row of Table 8) and generate a single ranking to which we apply the machine learning models.

Table 10 shows the results when only the full query is used to generate a ranked list of concepts. We again observe that SVM significantly outperforms J48, NB, and the baseline. mention false positives here: For both the J48 and NB classifiers we see a significant increase in precision (P1). Naive Bayes, for which precision was significantly worse on n-gram based reranking, performs significantly better than the other machine learning algorithms using full query reranking. The increase in precision comes at a loss in recall for NB. The MRR scores for J48 are no longer significantly higher than the baseline. Both J48 and NB produce fewer false positives when classifying full query data instead of n-gram based query data. This means that fewer incorrect concepts end up in the ranking which in turn results in a higher precision.

Interestingly, this increase in precision is not accompanied by a loss in recall. In particular, the SVM classifier is able to distinguish between correct and incorrect concepts when used on the full query data. 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 encouraging and shows that the approach taken handles the mapping of search engine queries to DBpedia extremely well.

7. Discussion

In this section, we further analyze the results presented in the previous section and answer the remaining research questions from Section 1. We first look at the inter-annotator agreement between the assessors. We then turn to the performance of the different textual representations of the Wikipedia content that we use. Further, we consider the robustness of the performance of our methods with respect to various parameter settings, provide an analysis of the influence of the feature types on the end results, and also report on the informativeness of the individual features. We conclude with an error analysis to see which queries are intrinsically difficult to map to the DBpedia portion of the LOD cloud.

Unless indicated otherwise, all results on which we report in this section use the best performing approach from the previous section, i.e., the SVM classifier with a linear kernel using the full queries (with ten-fold cross-validation when applicable).

7.1. Inter-annotator agreement

To assess the agreement between annotators, we randomly selected 50 sessions from the query log for judging by all annotators. We consider each query-concept pair to be an item of analysis for which each annotator expresses a judgment ("a good mapping" or "not a good mapping") and on which the annotators may or may not agree. However, our annotation tool does not produce any explicit labels of query-concept pairs as being "incorrect," since only positive ("correct") judgments are generated by the mappings. Determining the inter-annotator agreement on these positive judgments alone might bias the results and we adopt a modified approach to account for the missing non-relevance information, as we will now explain.

We follow the same setup as used for the results presented earlier by considering 5 concepts per query. In this case, the 5 concepts were sampled such that at least 3 were mapped (judged correct) by at least one of the annotators; the remaining concepts were randomly selected from the incorrect concepts. We deem a concept "incorrect" for a query if the query was not mapped to the concept by any annotator. For the queries where fewer than 3 correct concepts were identified, we increased the number of incorrect concepts to keep the total at 5. The rationale behind this approach is that each annotator looks at at least 5 concepts and selects the relevant ones. The measure of inter-annotator agreement that we are interested in is determined, then, on these 5 concepts per query. Also similar to the results reported earlier, we remove the queries in the "anomalous" categories.

The value for Cohen's κ is 0.5111, which indicates fair overall agreement (κ ranges from −1 for complete disagreement to +1 for complete agreement) [65–67]. Krippendorf's α is another statistic for measuring inter-annotator agreement that takes into account the probability that observed variability is due to chance. Moreover, it does not require that each annotator annotate each document [67,68]. The value of α is 0.5213. As with the κ value, this indicates a fair agreement between annotators. It is less, however, than the level recommended by Krippendorff for reliable data ($\alpha = 0.8$) or for tentative reliability ($\alpha = 0.667$). The values we obtain for α and κ are therefore an indication as to the nature of relevance with respect to our task. What one person deems a viable mapping given his or her background, another might find not relevant. Voorhees [69] has shown, however, that moderate inter-annotator agreement can still yield reliable comparisons between approaches (in her case TREC information retrieval runs, in our case different approaches to the mapping task) that are stable when one set of assessments is substituted for another. This means that, although the absolute inter-annotator scores indicate a fair agreement, the system results and comparisons thereof that we obtain are valid.

7.2. Textual concept representations

One of our baselines ranks concepts based on the full textual representation of each DBpedia concept, as described in Section 6.1. Instead of using the full text, we evaluate what the results are when we rank concepts based on each individual textual representation and based on combinations of fields. Table 11 lists the results. As per the Wikipedia authoring guidelines [53], the first sentence and paragraph should serve as an introduction to, and summary of, the important aspects of the contents of the article. In Table 11, we have also included these fields. From the table we observe that the anchor texts emerge as the best descriptor of each concept and using this field on its own obtains the highest absolute retrieval performance. However, the highest scores obtained using this approach are still significantly lower than the best performing machine learning method reported on earlier.

Table 11
Results of ranking concepts based on using the entire query **Q** using different textual representations of the Wikipedia article 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
Title	0.5651	0.5286	0.6523	0.6368	0.7363
Anchor	**0.6122**	**0.5676**	**0.7219**	**0.6922**	**0.8038**
First sentence	0.5495	0.5106	0.6523	0.6203	0.7268
First paragraph	0.5447	0.5048	0.6454	0.6159	0.7190
Title + content	0.5604	0.5200	0.6750	0.6357	0.7535
Title + anchor	0.5934	0.5621	0.7164	0.6792	0.7991
Title + content + anchor	0.5714	0.5302	0.6925	0.6514	0.7724
Title + 1st sentence + anchor	0.5856	0.5456	0.6965	0.6623	0.7755
Title + 1st paragraph + anchor	0.5777	0.5370	0.6985	0.6566	0.7771

7.3. Robustness

Next, we discuss the robustness of our approach. Specifically, we investigate the effects of varying the number of retrieved concepts in the first stage, of varying the size of the folds, of balancing the relative amount of positive and negative examples in the training data, and the effect of varying parameters in the machine learning models.

7.3.1. Number of concepts

The results in Section 6 were obtained by selecting the top 5 concepts from the first stage for each query, under the assumption that 5 concepts would give a good balance between recall and precision (motivated by the fact there are 1.34 concepts annotated per query on average). Our intuition was that, even if the initial stage did not place a relevant concept at rank 1, the concept selection stage could still consider this concept as a candidate (given that it appeared somewhere in the top 5). We now test this assumption by varying the number of concepts returned for each query.

Fig. 2 shows the effect of varying the number of retrieved concepts (K) in the first stage on various retrieval measures. On nearly all metrics the best performance is achieved when using the top 3 concepts from the initial stage for concept selection, although the absolute difference between using 3 and 5 terms is minimal for most measures. As we have observed above, most relevant concepts are already ranked very high by the initial stage. Further, from the figure we conclude that using only the top 1 is not enough and results in the worst performance. In general, one might expect recall to improve when the number of concepts grows. However, since each query only has 1.34 concepts annotated on average, recall can not improve much when considering larger numbers of candidate concepts. Finally, increasing the number of concepts mainly increases the number of non-relevant concepts in the training data, which may result in a bias towards classifying concepts as not relevant by a machine learning algorithm.

7.3.2. Balancing of the training set

Machine learning algorithms are sensitive to the distribution of positive and negative instances in the training set. The results reported so far do not perform any kind of resampling of the training data and take the distribution of the class labels (whether the current concept is selected by the assessors) as is.

In order to determine whether reducing the number of non-relevant concepts in the training data has a positive effect on the performance, we experiment using a balanced and a randomly distributed training set. The balanced set reduces the number of negative examples such that the training set contains as many positive examples as negative examples. On the other hand, the random sampled set follows the empirical distribution in the data. Table 12 shows that balancing the training set causes performance to drop. We thus conclude that including a larger number of negative examples has a positive effect on retrieval performance and that there is no need to perform any kind of balancing for our task.

7.3.3. Splitting the data

Ideally, the training set used to train the machine learning algorithms is large enough to learn a model of the data that is suffi-

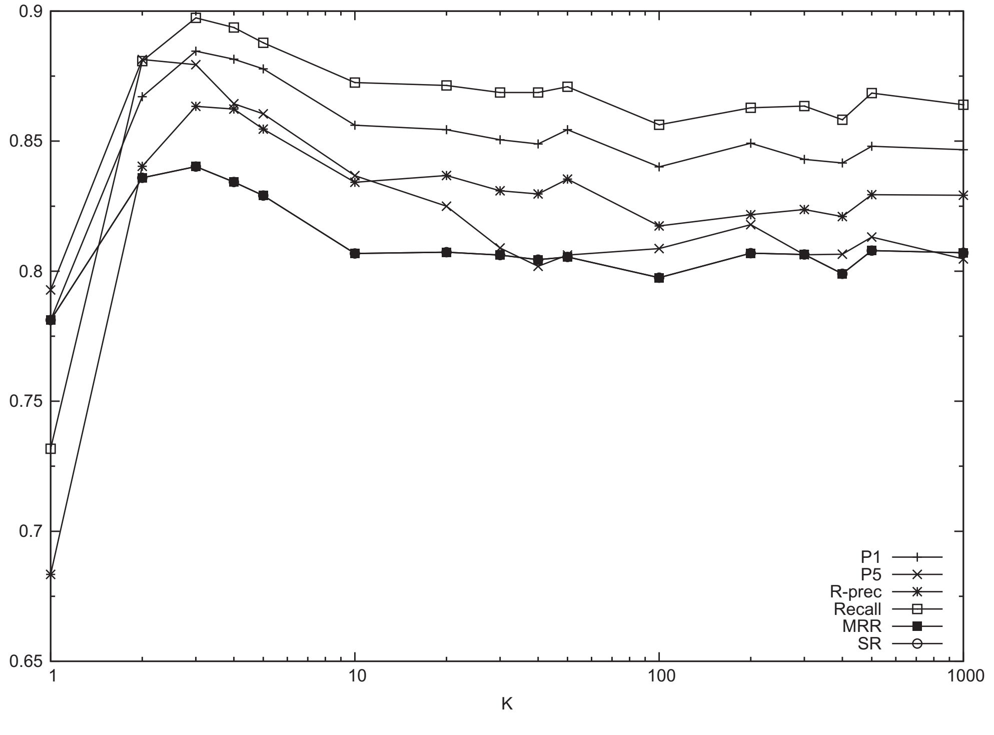


Fig. 2. Plot of results when varying the number of concepts K used as input to the concept selection stage on various evaluation measures. Note the log scale on the x-axis.

Table 12
Comparison of sampling methods for composing the training and test sets on the full queries.

	P1	R-prec	Recall	MRR	SR
Balanced	0.5777	0.4383	0.5436	0.5960	0.6150
Random sampling	**0.8833**	**0.8666**	**0.8975**	**0.8406**	**0.9053**

Table 13
Comparison of using different sizes for the training and test sets used for cross-validation. A 50–50 split uses the smallest training set (training and test set are equally sized), a 75–25 split uses 75% for training and 25% for testing, a 90–10 split uses 90% for training and 10% for testing.

	P1	R-prec	Recall	MRR	SR
50–50	0.8809	0.8601	0.8927	0.8338	0.9016
75–25	0.8812	0.8599	0.8927	0.8344	0.9015
90–10	**0.8833**	**0.8666**	**0.8975**	**0.8406**	**0.9053**

ciently discriminative; also, a test set should be large enough to test whether the model generalizes well to unseen instances.

Table 13 shows the results when we vary the size of the folds used for cross-validation using the SVM classifier on the full query based concept selection. Here, we compare the 90–10 split reported on above so far with a 50–50 and a 75–25 split. From this table we observe that there is only no significant difference between the results on various splits. In practical terms this means that the amount of training data can be greatly reduced, without a significant loss in performance. This in turn means that the labor-intensive, human effort of creating annotations can be limited to a few hundred annotations in order to achieve good performance.

Table 14
Comparison of using different kernels for the SVM machine learning algorithm.

	P1	R-prec	Recall	MRR	SR
Full query based concept selection					
Linear	**0.8833**	**0.8666**	**0.8975**	**0.8406**	**0.9053**
Gaussian	**0.8833**	**0.8666**	**0.8975**	**0.8406**	**0.9053**
Polynomial	0.8738	0.7859	0.8415	0.8364	0.8876
N-gram based concept selection					
Linear	0.7998	**0.6718**	0.7556	0.8131	0.8240
Gaussian	**0.8241**	0.6655	**0.7849**	**0.8316**	**0.8641**
Polynomial	0.7967	0.6251	0.7660	0.8205	0.8589

7.3.4. Machine learning model parameters

Next, we look at important parameters of the three machine learning algorithms we evaluate.

Table 14 shows the results of using different kernels for the SVM classifier, specifically a linear, a gaussian, and a polynomial kernel. On the full query data there is no difference between the linear and gaussian kernel and on the n-gram data there is only a small difference. The polynomial kernel performs the worst in both cases, but again the difference is insignificant as compared to the results attained using the other kernels. The values listed in Table 14 are obtained using the optimal parameter settings for the kernels. Fig. 3b shows a sweep of the complexity parameter for the gaussian kernel. A higher degree of complexity penalizes non-separable points and leads to overfitting, while if the value is too low SVM is unable to learn a discriminative model. For the polynomial kernel we limited our experiments to a second order kernel, as the increase in training times on higher order kernels made further experimentation prohibitive. The fact that there is little difference between the results of using various kernels shows that, for the purpose of reranking queries, a simple linear model is enough to achieve optimal or close to optimal performance. A more complex model leads to limited or no improvement and increased training times.

Table 15 shows the results of binning versus kernel density estimation (using a gaussian kernel). As was the case with SVM, there is only a small difference between the results on the full query data. The results on the n-gram data do show a difference; binning performs better in terms of recall while kernel density estimation achieves higher precision, which is probably caused by the kernel method overfitting the data.

Fig. 3a shows the effect of varying the level of pruning for the J48 algorithm on the full query data, where a low number relates to more aggressive pruning. We observe that more agressive pruning leads to slightly better performance over the standard level (0.25), but not significantly so.

An exploration of the machine learning model parameters shows that SVM is the best classifier for our task. Even with optimized parameters the Naive Bayes and J48 classifiers do not achieve better results.

7.4. Feature types

In Section 4.3 we identified four 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. To this end we perform an ablation experiment, where each of these groups is removed from the training data.

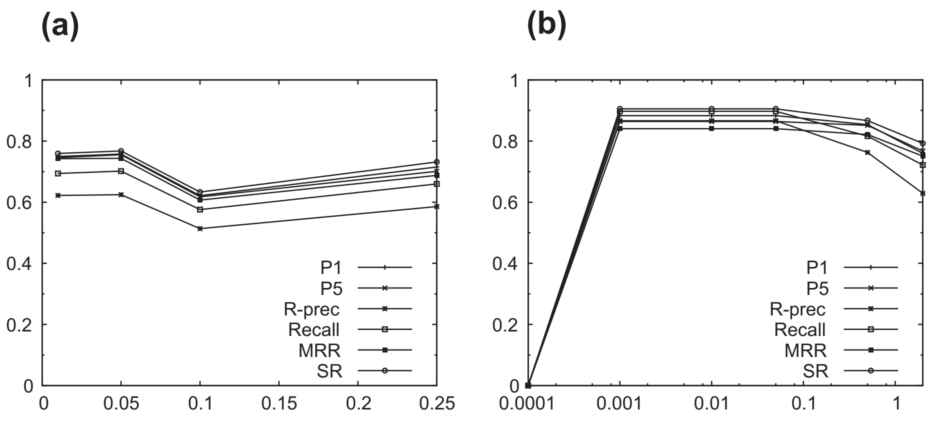


Fig. 3. (a) The effect of adjusting the pruning parameter for the J48 learning algorithm. A lower number means more aggressive pruning. (b) The effect of adjusting the complexity parameter for SVM with a gaussian kernel. Note that the *x*-axis is on a log scale.

Table 15
Comparison of using different probability density estimation methods for the NB classifier.

	P1	R-prec	Recall	MRR	SR
Full query based concept selection					
Binning	**0.6925**	0.5897	0.6865	**0.6989**	**0.7626**
Kernel	0.6897	**0.5973**	**0.6882**	0.6836	0.7455
N-gram based concept selection					
Binning	0.4494	**0.4088**	**0.6948**	**0.7278**	**0.7710**
Kernel	**0.5944**	0.3236	0.4884	0.5946	0.6445

Table 16
Results of removing specific feature types from the training data for the SVM classifier and n-gram based concept selection. ▼ 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.

Excluded feature types	P1	R-prec	Recall	MRR	SR
–	**0.7998**	**0.6718**	**0.7556**	**0.8131**	**0.8240**
History	0.6848°	0.5600°	0.6285°	0.6902°	0.6957°
Concept	$0.4844^{\circ\circ}$	$0.3895^{\blacktriangledown\circ}$	$0.4383^{\blacktriangledown\circ}$	$0.4875^{\blacktriangledown\circ}$	$0.4906^{\blacktriangledown\circ}$
History; concept	$0.2233^{\blacktriangledown\blacktriangledown\circ}$	$0.1233^{\blacktriangledown\blacktriangledown\circ}$	$0.1733^{\blacktriangledown\blacktriangledown\circ}$	$0.2233^{\blacktriangledown\blacktriangledown\circ}$	$0.2233^{\blacktriangledown\blacktriangledown\circ}$

Table 18
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-gram based concept selection		Full query based concept selection	
0.119	$RANK(c,Q)$	0.190	$RANK(c,Q)$
0.107	ID	0.108	$TEQ(c,Q)$
0.052	$INLINKS(c)$	0.080	$INLINKS(c)$
0.040	$TF_{anchor(c,Q)}$	0.056	ID
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	$REDIRECT(c)$

7.4.1. N-gram based concept selection

Table 16 shows the results using n-gram based concept selection. It turns out that both the n-gram specific and n-gram + concept specific features are required for successful classification. When these groups are removed, none of the relevant concepts are identified. From Table 16 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. So, although some feature types contribute more to the final performance, each is needed to arrive at the highest scores.

7.4.2. Full-query based concept selection

Table 17 shows the results of using full-query based concept selection. In this case, the effect of removing both history and concept based features does not influence the results at all. This can in part be explained by the fact that most history features are based on the counts of the query in various parts of the session. Since we now have a single n-gram (the full query), these counts turn into binary features and may therefore be less discriminative. This is in stark contrast with the n-gram based features that do have a significant effect on performance on all metrics. Similar to the n-gram based data, these features are essential for full query based concept selection. Finally, we observe that there are some dependencies among the types of features. When we remove both the n-gram + concept features and the history features, the performance is worse than when we remove only the n-gram + concept features (although not significantly so).

7.4.3. Upshot

In sum, all feature types contribute to the performance in the case of n-gram based concept selection. The highest scores are obtained, however, using full query based concept selection. In this case, the history and concept based features do not contribute significantly to the results.

7.5. Feature selection

Several methods exist for automatically determining the most informative features given training instances and their class labels. In this section we report on using an information gain based algorithm for feature selection [70].

Table 18 lists the features with the highest information gain values for both n-gram and full query based reranking. 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 good indicators of the relevance status of a concept. *ID* 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 our manually created assessments 854 concepts are identified, 505 of which are unique; some of the repetitions are caused because of a persisting 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 concept selection, the term frequency in the title and anchor texts are strong indicators of relevance for given phrase and concept.

7.6. Error analysis

Finally, we provide an analysis of the errors that were made by the machine learning algorithms. To this end, we first examine the relationship between mapping performance and the fre-

Table 17
Results of removing specific feature types from the training data for the SVM classifier and full query based concept selection. Not all possible combinations are included in the results; all unlisted combinations have either scores of zero or the same score as when using all feature types. ▼ 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 n-gram + concept and n-gram features run, and the rightmost symbol the without n-gram + concept features run.

Excluded feature types	P1	R-prec	Recall	MRR	SR
–	**0.8833**	**0.8666**	**0.8975**	**0.8406**	**0.9053**
History; concept	$\mathbf{0.8833}^{\circ}$	$\mathbf{0.8666}^{\circ}$	$\mathbf{0.8975}^{\circ}$	$\mathbf{0.8406}^{\circ}$	$\mathbf{0.9053}^{\circ}$
N-gram + concept; n-gram	$0.1000^{\blacktriangledown}$	$0.0000^{\blacktriangledown}$	$0.0500^{\blacktriangledown}$	$0.1000^{\blacktriangledown}$	$0.1000^{\blacktriangledown}$
N-gram + concept	$0.0556^{\blacktriangledown\circ}$	$0.0222^{\blacktriangledown\circ}$	$0.0370^{\blacktriangledown\circ}$	$0.0556^{\blacktriangledown\circ}$	$0.0556^{\blacktriangledown\circ}$
N-gram + concept; history	$0.0333^{\blacktriangledown\circ\circ}$	$0.0000^{\blacktriangledown\circ\circ}$	$0.0167^{\blacktriangledown\circ\circ}$	$0.0333^{\blacktriangledown\circ\circ}$	$0.0333^{\blacktriangledown\circ\circ}$

quency of the query in the entire query log. We separate all queries in two groups; one for those queries where our approach successfully mapped concepts and one where it failed. In the first group, the average query frequency is 23 (median 2, std. dev. 85.6). In the second group, the average frequency is 6 (median 1, std. dev. 19.5). So, although it seems our approach works best for frequently occurring queries, the high standard deviation indicates that the frequencies are spread out over a large range of values.

Table 19 shows examples of correctly and incorrectly mapped queries, together with their relative frequency of occurrence in the entire query log. This table provides further indication that the frequency of a query is not a determining factor in the successful outcome of our method. Rather, it is the retrieval framework that puts concepts that contain query terms with a relatively high frequency in the top of the ranking. For example, Beatrix is, besides the queen of the Netherlands, also the name of one of the characters in the movie Kill Bill. The best results are obtained when the machine learning algorithm selects the right concepts from the initial ranking.

To further investigate the errors being made, we have manually inspected the output of the algorithms and classified the errors into several classes. Since we formulate our task as a ranking problem, we are primarily interested in the false positives—these are the concepts the classifier identified as correct for a query but which the annotators did not select. The classes in which the classifiers make the most mistakes are:

- *Ambiguous (7%):* A query may map to more than one concept and the annotators did not explicitly mark the query as being ambiguous.
- *Match with term in content (16%):* Part of the query occurs frequently in the textual representation of the concept, while the concept itself is not relevant. For example, the query "red lobster" matches with the concept RED CROSS.
- *Substring (5%):* In this case a substring of the query is matched to a concept, for example the concept BROOKLYN is selected for the query "brooklyn bridge." While this might be considered an interesting suggestion, it is incorrect since the annotators did not label it so.
- *Too specific—child selected (12%):* A narrower concept is selected where the broader is correct. For example, when the concept EUROVISION SONGFESTIVAL 1975 is selected for the query "songfestival."
- *Too broad—parent selected (7%):* The inverse of the previous case. For example, the concept EUROVISION is selected for the query "eurovision songfestival 2008."
- *Related (12%):* A related concept is selected. For example when the concept CUBA CRISIS is selected for the query "cuba kennedy." Another example is the concept INDUSTRIAL DESIGN for the query "walking frame."
- *Sibling (5%):* A sibling is selected, e.g., EUROVISION SONGFESTIVAL 1975 instead of EUROVISION SONGFESTIVAL 2008.
- *Same concept, different label (7%):* When there is more than one applicable concept for the query and the annotators used only one, e.g., in the case of NEW YORK and NEW YORK CITY.
- *Erroneous (28%):* The final category is where the classifiers selected the right concept, but it was missed by the annotators.

Table 19
Examples of correctly and incorrectly mapped queries, with their relative frequency of occurrence in the entire query log. Concepts annotated by the human annotators in boldface. Wouter Bos is a Dutch politician and Beatrix is the Dutch queen.

Freq. ($\times 10^{-4}$)	Query	Mapped concepts
Good performing queries		
64.0%	wouter bos	**WOUTER BOS**
18.9%	moon landing	**MOON LANDING**
2.22%	vietnam war	**VIETNAM WAR**
1.67%	simple minds	**SIMPLE MINDS**
1.11%	spoetnik	**SPOETNIK**
1.11%	sarkozy agriculture	**NICOLAS SARKOZY**; AGRICULTURE
0.557%	universal soldier	**UNIVERSAL SOLDIER**
Bad performing queries		
57.9%	gaza	DOROTHEUS OF GAZA
2.78%	wedding beatrix	KILL BILL; WILLEM OF LUXEMBURG; MASAKO OWADA
1.11%	poverty netherlands 1940s	**1940-1949**; IMMIGRATION POLICY; MEXICAN MIRACLE
0.557%	poverty thirties	**1930-1939**; HUMAN DEVELOPMENT INDEX
0.557%	rabin funeral	BILL CLINTON; HUSSEIN OF JORDAN
0.557%	eurovision songfestival 1975	EUROVISION SONGFESTIVAL; MELODIFESTIVALEN 1975
0.557%	cold war netherlands	**COLD WAR**; WATCHTOWER; WESTERN BLOC

From these classes we conclude that the largest part of the errors are not attributable to the machine learning algorithms but rather to incomplete human annotations. Another class of interesting errors is related to the IR framework we use. This sometimes produces "fuzzy" matches when the textual representation of the concept contains the query terms with a high frequency (e.g., selecting CUBA CRISIS for the query "cuba kennedy"). We argue that some of these and other errors are not wrong per se, but interesting since they do provide mappings to related concepts. Marking them as wrong is partly an artifact of our evaluation methodology, which determines a priori which concepts are relevant to which queries, so as to ensure the reusability our evaluation resources. We have chosen this approach also for practical reasons, since the same annotations are used to generate the training data for the machine learners. In future work, we intend to perform a large-scale post-hoc evaluation in which we directly evaluate the generated mappings.

8. Conclusion and future work

We have introduced the task of mapping search engine queries to the LOD cloud and presented a method that uses supervised machine learning methods to learn which concepts are used in a query. We consider DBpedia to be an integral part of, and interlinking hub for, the LOD cloud, which is why we focused our efforts on mapping queries to this ontology. The concepts suggested by our method may be used to provide contextual information, related concepts, or navigational suggestions to the user although they could also simply be used as an entry point into the Linking Open Data cloud. Our approach first retrieves and ranks candidate concepts using a framework based on language modeling for information retrieval. We then extract query, concept, and history-specific feature vectors for these candidate concepts. Using manually created annotations we inform a machine learning algorithm, which then learns how to best select candidate concepts given an input query. We found that simply performing a lexical match between the queries and concepts did not perform well and neither did using retrieval alone, i.e., omitting the concept selection stage. When applying our proposed method, we found significant improvements over these baselines.

Our best performance was obtained using Support Vector Machines and features extracted from the full input queries. The best performing run was able to locate almost 90% of the relevant concepts on average. Moreover, this particular run achieved a precision@1 of 89%, meaning that for this percentage of queries the first suggested concept was relevant.[4] With respect to the machine learning algorithms, we found that reducing the quantity of training material caused only a marginal decline in performance. This means, in practical terms, that the amount of labor-intensive human annotations can be greatly reduced.

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 many of the employed features can be used with ontologies other than DBpedia. However, as became clear from Table 16 and 18, DBpedia related features such as inlinks and outlinks and redirects were helpful. We also found that features pertaining to both the concept and query (such as the term frequency of the query in various textual representations of the concepts) were essential in obtaining good classification performance. Such information may not exist in other ontologies.

In sum, we have shown that search engine queries can be successfully mapped to DBpedia concepts. In our evaluation, the best approach incorporated both information retrieval and machine learning techniques. The best way of handling query terms is to model them as a single unit—a finding also interesting from an efficiency viewpoint, since the number of n-grams is quadratic in the length of the query.

We identify a number of points to be addressed in future work. First, as is inherent to real-world logged data, the queries are specific to the given system and domain. This raises questions about the generalizability of the results to other, broader domains. In another paper, we have applied the same approach to query sets taken from the TREC evaluation campaign, including a set taken from a commercial web search engine's query log [71]. The end goal in that paper was different, namely to use the found mappings to improve end-to-end document retrieval. The results, however, were comparable to the results presented in this paper. We leave other cross-domain investigations for future work.

We also plan to experiment with additional features, for example by including more structural ones such as those pertaining to the structure of the ontology. Although we have found that our current method obtained convincing results and improvements over the two baselines, we believe that further improvements can be obtained by considering the graph structure of DBpedia (or the LOD cloud in general). One example of such an improvement could be to use the graph structure to "zoom in" on a relevant subgraph of the knowledge repository. This information could then be used to determine the concepts closest to or contained in this graph.

One other aspect that we intend to investigate in the future is how to incorporate information from, or link queries to other parts of the LOD cloud. Our current approach has focused on DBpedia, which might be too limited for some queries [72]. As indicated in Section 2, it is common in the field of natural language interfaces to databases to find matches between keywords from the query and records in a database. Once these are found, all records that can be joined together are returned as a single result. A similar approach is obviously also possible using the LOD cloud and future work should indicate whether traversing links to other, connected knowledge repositories would yield additional relevant concepts and/or relations.

Furthermore, the generated mappings between queries and concepts can be interpreted as user-generated tags or alternative labels for the concepts. As such, they might be used to discover or confirm aspects of the concepts. We also intend to go beyond suggesting concepts and look at which part of the query should be linked. There might be room for further improvements by using session history in other ways. One option would be a more fine-grained notion of session changes, e.g., using query overlap [73]. Another option would be to include more context by considering the user history over multiple sessions.

Finally, our task definition requires fairly strict matches between the search engine queries and DBpedia concepts, comparable to finding `skos:exactMatch` or even `owl:equivalentClass` relations in an ontology matching task. However, our task can also be interpreted in a broader sense, where not only exact matches but also semantically related concepts are suggested [72,74]. For example, when a query contains a book title that is not represented by a concept in DBpedia, we could suggest its author (assuming the book title is mentioned in the author's Wikipedia page). Similarly, instances for which no concept is found can be linked to a more general concept. We believe that our approach can be adapted to incorporate such semantically related general instances of a specific concept could be defined as a correct concept for mapping.

Acknowledgments

This research was supported by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement number 250430, the PROMISE Network of Excellence co-funded by the 7th Framework Programme of the European Commission, grant agreement number 258191, the DuOMAn project carried out within the STEVIN Programme which is funded by the Dutch and Flemish Governments under project number STE-09-12, the Netherlands Organization for Scientific Research (NWO) under project numbers 612.061.814, 612.061.815, 640.004.802, 380.70.011, the Center for Creation, Content and Technology (CCCT), the Dutch Ministry of Economic Affairs and Amsterdam Topstad under the Krant van Morgen project, the Hyperlocal Service Platform project funded by the Service Innovation & ICT program, and the WAHSP project funded by the CLARIN-nl program.

References

[1] E.J. Meij, M. Bron, B. Huurnink, L. Hollink, M. de Rijke, Learning semantic query suggestions, in: ISWC '09: Proceedings of the 8th International Semantic Web Conference, 2009, pp. 424–440.
[2] T. Berners-Lee, Linked Data—Design Issues, 2009. Available from: <http://www.w3.org/DesignIssues/LinkedData.html>.
[3] C. Bizer, T. Heath, K. Idehen, T. Berners-Lee, Linked data on the web (LDOW2008), in: WWW '08: Proceeding of the 17th International Conference on World Wide Web, 2008, pp. 1265–1266.
[4] C. Bizer, T. Health, T. Berners-Lee, Linked data—the story so far, International Journal on Semantic Web and Information Systems (IJSWIS) 5 (3) (2009) 1–22.
[5] F.M. Suchanek, G. Kasneci, G. Weikum, YAGO: a large ontology from Wikipedia and WordNet, Web Semantics: Science, Services and Agents on the World Wide Web 6 (3) (2008) 203–217.
[6] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives, DBpedia: a nucleus for a web of open data, in: Proceedings of 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC + ASWC 2007), 2007, pp. 722–735.
[7] M. Banko, M.J. Cafarella, S. Soderland, M. Broadhead, O. Etzioni, Open information extraction from the web, in: M.M. Veloso (Ed.), IJCAI, 2007, pp. 2670–2676.
[8] W.R. van Hage, M. de Rijke, M. Marx, Information retrieval support for ontology construction and use, in: ISWC '04: Proceedings of the 3rd International Semantic Web Conference, 2004, pp. 518–533.
[9] R. Mihalcea, A. Csomai, Wikify!: linking documents to encyclopedic knowledge, in: CIKM '07: Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, 2007, pp. 233–242.

[4] Our results 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 or vague for human assessors to be able to assign a concept to. Ideally, one would have a classifier at the very start of the query linking process which would predict whether a query falls in one of these categories. Implementing and evaluating such a classifier is an interesting—and challenging—research topic in itself and falls beyond the scope of our current work.

[10] D. Milne, I.H. Witten, Learning to link with Wikipedia, in: CIKM '08: Proceedings of the 17th ACM Conference on Information and Knowledge Management, 2008, pp. 509–518.
[11] B.J. Jansen, A. Spink, 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) (2000) 207–227.
[12] P. Anick, R.G. Kantamneni, A longitudinal study of real-time search assistance adoption, in: SIGIR '08: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2008, pp. 701–702.
[13] E. Meij, P.Mika, H. Zaragoza, An evaluation of entity and frequency based query completion methods, in: SIGIR '09: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009, pp. 678–679.
[14] D. Blocks, C. Binding, D. Cunliffe, D. Tudhope, Qualitative evaluation of thesaurus-based retrieval, in: ECDL '02: Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries, 2002, pp. 346–361.
[15] T. Tsikrika, C. Diou, A. de Vries, A. Delopoulos, Image annotation using clickthrough data, in: CIVR '09: Proceeding of the ACM International Conference on Image and Video Retrieval, 2009, pp. 1–8.
[16] P. Mika, E. Meij, H. Zaragoza, Investigating the semantic gap through query log analysis, in: ISWC '09: Proceedings of the 8th International Semantic Web Conference, 2009, pp. 441–455.
[17] A. Spink, B.J. Jansen, D. Wolfram, T. Saracevic, From E-sex to E-commerce: web search changes, IEEE Computer 35 (3) (2002) 107–109.
[18] J.X. Yu, L. Qin, L. Chang, Keyword search in relational databases: a survey, IEEE Data Engineering Bulletin 33 (1) (2010) 67–78 (Special Issue on Keyword Search).
[19] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, S. Sudarshan, Keyword searching and browsing in databases using BANKS, in: Proceedings of the 18th International Conference on Data Engineering, 2002, pp. 431–440.
[20] V. Hristidis, Y. Papakonstantinou, DISCOVER: Keyword Search in Relational Databases, in: VLDB, Morgan Kaufmann, 2002, pp. 670–681.
[21] S. Agrawal, S. Chaudhuri, G. Das, DBXplorer: a system for keyword-based search over relational databases, in: Proceedings of the 18th International Conference on Data Engineering, 2002, pp. 5–16.
[22] S. Tata, G.M. Lohman, SQAK: doing more with keywords, in: SIGMOD Conference, ACM, 2008, pp. 889–902.
[23] G. Kasneci, F.M. Suchanek, G. Ifrim, M. Ramanath, G. Weikum, NAGA: searching and ranking knowledge, in: ICDE, IEEE, 2008, pp. 953–962.
[24] S. Elbassuoni, M. Ramanath, R. Schenkel, M. Sydow, G. Weikum, Language-model-based ranking for queries on RDF-graphs, in: CIKM '09: Proceeding of the 18th ACM Conference on Information and Knowledge Management, ACM, 2009, pp. 977–986.
[25] E. Demidova, P. Fankhauser, X. Zhou, W. Nejdl, DivQ: diversification for keyword search over structured databases, in: SIGIR '10: Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2010, pp. 331–338.
[26] E. Kaufmann, A. Bernstein, Evaluating the usability of natural language query languages and interfaces to semantic web knowledge bases, Web Semantics: Science, Services and Agents on the World Wide Web 8 (2010) 377–393.
[27] C. Caracciolo, J. Euzenat, L. Hollink, R. Ichise, A. Isaac, V. Malaisé, C. Meilicke, J. Pane, P. Shvaiko, H. Stuckenschmidt, O. Šváb, V. Svátek, Results of the ontology alignment evaluation initiative 2008, in: The Third International Workshop on Ontology Matching at ISWC, 2008, pp. 73–120.
[28] P. Shvaiko, J. Euzenat, A survey of schema-based matching approaches, Journal on Data Semantics 4 (3730) (2005) 146–171.
[29] S. Wang, G. Englebienne, S. Schlobach, Learning concept mappings from instance similarity, in: ISWC '08: Proceedings of the 7th International Semantic Web Conference, 2008, pp. 339–355.
[30] G. Stoilos, G.B. Stamou, S.D. Kollias, A string metric for ontology alignment, in: ISWC '05: Proceedings of the 4th International Semantic Web Conference, 2005, pp. 624–637.
[31] Z. Aleksovski, M.C.A. Klein, W. ten Kate, F. van Harmelen, Matching unstructured vocabularies using a background ontology, in: Managing Knowledge in a World of Networks, 15th International Conference, EKAW, 2006, pp. 182–197.
[32] P. Buitelaar, P. Cimiano, B. Magnini, Ontology Learning from Text: Methods, Evaluation and Applications, IOS Press, 2005.
[33] B. Fortuna, M. Grobelnik, D. Mladenic, OntoGen: semi-automatic ontology editor, in: Proceedings of the 2007 Conference on Human Interface, 2007, pp. 309–318.
[34] A. Maedche, R. Volz, The ontology extraction maintenance framework text-to-onto, in: Proceedings of the IEEE International Conference on Data Mining.
[35] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. Tomlin, J. Zien, SemTag and seeker: bootstrapping the semantic web via automated semantic annotation, in: Proceedings of the 12th International Conference on World Wide Web, 2003, pp. 178–186.
[36] V. Malaisé, L. Gazendam, H. Brugman, Disambiguating automatic semantic annotation based on a thesaurus structure, in: TALN 2007: Actes de la 14e conférence sur le Traitement Automatique des Langues Naturelles.
[37] B. Huurnink, L. Hollink, W. van den Heuvel, M. de Rijke, The search behavior of media professionals at an audiovisual archive: a transaction log analysis, Journal of the American Society for Information Science and Technology 61 (2010) 1180–1197.
[38] A. Kiryakov, B. Popov, I. Terziev, D. Manov, D. Ognyanoff, Semantic annotation, indexing, and retrieval, Web Semantics: Science, Services and Agents on the World Wide Web 2 (1) (2004) 49–79.
[39] B. Popov, A. Kiryakov, D. Manov, A. Kirilov, D. Ognyanoff, M. Goranov, Towards semantic web information extraction, in: Human Language Technologies Workshop at the 2nd International Semantic Web Conference (ISWC2003), 2003, pp. 2–22.
[40] A. Bhole, B. Fortuna, M. Grobelnik, D. Mladenic, Extracting named entities and relating them over time based on Wikipedia, Informatica 4 (4) (2007) 463–468.
[41] C. Chemudugunta, A. Holloway, P. Smyth, M. Steyvers, Modeling documents by combining semantic concepts with unsupervised statistical learning, in: ISWC '08: Proceedings of the 7th International Semantic Web Conference, 2008, pp. 229–244.
[42] C. Fellbaum, M. Palmer, H.T. Dang, L. Delfs, S. Wolf, Manual and automatic semantic annotation with WordNet, in: WordNet and Other Lexical Resources, 2001, pp. 3–10.
[43] J. Guo, G. Xu, X. Cheng, H. Li, Named entity recognition in query, in: SIGIR '09: 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009, pp. 267–274.
[44] G. Mishne, M. de Rijke, A study of blog search, in: ECIR '06: Proceedings of the 28th European Conference on Information Retrieval, 2006, pp. 289–301.
[45] S.M. Beitzel, E.C. Jensen, A. Chowdhury, D. Grossman, O. Frieder, Hourly analysis of a very large topically categorized web query log, in: SIGIR '04: 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2004, pp. 321–328.
[46] B.J. Jansen, A. Goodrum, A. Spink, Searching for multimedia: analysis of audio, video and image Web queries, World Wide Web 3 (4) (2000) 249–254.
[47] B. Huurnink, L. Hollink, W. vanden Heuvel, M. de Rijke, Search behavior of media professionals at an audiovisual archive: a transaction log analysis, Journal of the American Society for Information Science and Technology 61 (6) (2010) 1180–1197.
[48] L. Mihalkova, R. Mooney, Learning to disambiguate search queries from short sessions, in: ECML PKDD '09: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 111–127, ISBN 978-3-642-04173-0.
[49] Z. Dou, R. Song, J.-R. Wen, A large-sc ale evaluation and analysis of personalized search strategies, in: WWW '07: Proceedings of the 16th International Conference on World Wide Web, ACM, New York, NY, USA, 2007, pp. 581–590, ISBN 978-1-59593-654-7.
[50] M. Bendersky, W.B. Croft, Discovering key concepts in verbose queries, in: SIGIR '08: 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2008, pp. 491–498.
[51] Y. Zhou, B.W. Croft, Query performance prediction in web search environments, in: SIGIR '07: 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2007, pp. 543–550.
[52] M. Bendersky, W.B. Croft, Analysis of long queries in a large scale search log, in: WSCD '09: Proceedings of the 2009 Workshop on Web Search Click Data, 2009, pp. 8–14.
[53] Wikipedia: Lead Section, 2010. Available from: <http://en.wikipedia.org/wiki/Wikipedia:Lead_section>.
[54] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, Morgan Kaufmann, 2005.
[55] V.N. Vapnik, The Nature of Statistical Learning Theory, Springer-Verlag, 1995.
[56] D. Hiemstra, Using Language Models for Information Retrieval, Ph.D. thesis, University of Twente, 2001.
[57] J.M. Ponte, W.B. Croft, A language modeling approach to information retrieval, in: SIGIR '98: 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1998, pp. 275–281.
[58] C. Zhai, J. Lafferty, A study of smoothing methods for language models applied to information retrieval, ACM Transactions on Information Systems 22 (2) (2004) 179–214.
[59] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Addison Wesley, 1999.
[60] K.W. Church, W.A. Gale, Inverse document frequency (IDF): a measure of deviations from Poisson, in: Proceedings of the Third Workshop on Very Large Corpora, 1995, pp. 121–130.
[61] M.D. Smucker, C.L.A. Clarke, G.V. Cormack, Experiments with ClueWeb09: relevance feedback and web tracks, in: Proceedings of the 18th Text Retrieval Conference (TREC 2009).
[62] R.J. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, 1993.
[63] G.H. John, P. Langley, Estimating continuous distributions in Bayesian classifiers, in: UAI '95: Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence, 1995, pp. 338–345.
[64] J.C. Platt, Fast training of support vector machines using sequential minimal optimization, in: Advances in Kernel Methods: Support Vector Learning, MIT Press, 1999, pp. 185–208.
[65] J. Cohen, A coefficient of agreement for nominal scales, Educational and Psychological Measurement 20 (1) (1960) 37–46.
[66] R.J. Landis, G.G. Koch, The measurement of observer agreement for categorical data, Biometrics 33 (1) (1977) 159–174.
[67] R. Artstein, M. Poesio, Inter-coder agreement for computational linguistics, Computational Linguistics 34 (4) (2008) 555–596.
[68] A. Hayes, K. Krippendorf, Answering the call for a standard reliability measure for coding data, Communication Methods and Measures 1 (1) (2007) 77–89.
[69] E.M. Voorhees, Variations in relevance judgments and the measurement of retrieval effectiveness, Information Processing & Management 36 (5) (2000) 697–716.

[70] Y. Yang, J.O. Pedersen, A comparative study on feature selection in text categorization, in: ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning, 1997, pp. 412–420.
[71] E. Meij, M. de Rijke, Supervised query modeling using Wikipedia, in: Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '10, ACM, 2010.
[72] K. Balog, E. Meij, M. de Rijke, Entity search: building bridges between two worlds, in: Proceedings of the Workshop on Semantic Search (SemSearch 2010) at the 19th International World Wide Web Conference (WWW 2010), 2010.
[73] B.J. Jansen, A. Spink, C. Blakely, S. Koshman, Defining a session on Web search engines, Journal of the American Society for Information Science and Technology 58 (6) (2007) 862–871.
[74] E. Meij, P. Mika, H. Zaragoza, Investigating the demand side of semantic search through query log analysis, in: Proceedings of the Workshop on Semantic Search (SemSearch 2009) at the 18th International World Wide Web Conference (WWW 2009), 2009.