

# A Survey of Semantic Search Approaches

Thanh Tran <sup>#</sup>, Peter Mika <sup>†</sup>

<sup>#</sup>*Institute AIFB,  
Karlsruhe Institute of Technology, Germany*  
duc.tran@kit.de

<sup>†</sup>*Yahoo! Research,  
Barcelona, Spain*  
pmika@yahoo-inc.com

**Abstract**—The use of semantics for various search tasks and the topic known as semantic search formed around it have attracted attention from researchers in many different communities and stirred up commercial expectations and interests. Because this topic is studied by different communities from different viewpoints, a wide range of semantic search solutions have been proposed. They greatly vary in the data, the semantic resources, the information needs and the querying paradigms that are supported, as well as the many search subproblems they focus on, from the interpretation of query inputs and data, to matching the query intent against data and up to ranking results. Along these aspects, we present a systematic survey to provide a taxonomy of various semantic search approaches, to reflect on the achievements that have been made over these years, and finally, to discuss the challenges ahead and directions for future research.

**Index Terms**—Search, semantic Search, data retrieval, semantic data retrieval.



## 1 INTRODUCTION

Advancements in search technologies enable users to deal with and to exploit the ever increasing amount of information that we can find in different settings – from the personal desktop to enterprises’ databases up to the large-scale Web. As opposed to the structured querying capabilities commonly provided by commercial databases that are only accessible by experts, *search* is rather understood as an end-user oriented paradigm that is based on intuitive interfaces and access mechanisms. Solutions that are widely used especially on the Web are based on keyword search, and in smaller-scale scenarios, also support natural language (NL) inputs. While they are easy to use, the inputs provided by the users through these interfaces are *ambiguous* such that the underlying information need (query intent) is not directly accessible to the system. Further, there are search scenarios where not only the query, but also the data representation does not precisely capture the semantics and structure of the underlying information. Accordingly, the core problems to be addressed in search are (1) to *understand the query intent and the data*, (2) to *match query against data*, and (3) to *rank results*.

Modern search engines are efficient in retrieving documents (web pages in the case of web search) from large collections based on matching the text of the user’s query with the textual content of the document or words in linking to the current document (*anchortext*). These textual features are often combined with additional features that measure the popularity of results based on usage data (*click-behaviour* or using metrics computed on the web graph such as *PageRank* and its derivatives). In combination, these relatively simple, but scalable matching techniques are able to address a large fraction of common information

needs. They work particularly well when the user is able to explicitly name the target of the query, this reference is unambiguous and refers to a popular item widely referred to with that name. An example is a product search, where the user enters the name of the product and that product is widely known by that name, e.g. *canon ixus 115*, a digital camera.

### 1.1 Many Unsolved Long-tail Queries

While search engines have been effective in addressing common needs, most queries are still in the “long tail” of Web query logs. Baeza-Yates et al. report that even in a full year of query logs 88% of the unique queries are singleton queries, i.e. appeared only once in the year. A lot of this variety comes from syntactic variations of the same popular need and can be addressed with spell-corrections and query suggestion techniques. However, the remaining queries tend to be less common because they refer to less popular needs that need disambiguation or require more complex explanation.

**Ambiguous Queries:** Ambiguous queries arise naturally when the user may not be able to name precisely the item that is sought. Coming back to the example of product search, a user who is in the early stages of selection may not know exactly which product to look for. In this case, the user may name one or two characteristics of the product, such as *cheap digital camera*. The concept of digital camera covers a broad range of actual entities, and there might be some discussion as to which of these can be considered cheap. There are also systemic factors that lead to an increase in ambiguity. Despite the expectation that longer queries should leave to better results, current search engines perform poorer on longer specifications of

the same information need, largely because search engines employ an AND semantics between words, and the space of documents that contain all the words becomes smaller with every word added and determining the important words becomes more difficult. Thus users have learned to avoid long queries, leading to shorter but more ambiguous queries. Lastly, users will try to find what they are looking for with the least possible words, adding additional terms after realizing that their original query was ambiguous. For example, when searching for people the user may type in a name, only to realize that there are multiple persons with the same name. Often, this is aggravated by the bias toward a single popular answer. For example, when searching for 'George Bush', the top results are crowded by references to the same person, even though this is a common name.<sup>1</sup>

**Complex Queries:** We include in this class all queries that go beyond a reference to a single named entity, and involve several entities and relationships between them. These queries arise again when the user does not know the answer to a question and therefore has to describe the item sought and its relationships to other items that he or she can name. For example, searching for this paper by the name of the second author is relatively straightforward. However, assuming the user does not know (or do not remember) the name, he or she might look for a "semantic search survey paper written by a researcher at yahoo". Complex queries are particularly hard to address in the traditional information retrieval paradigm when they require information extraction and aggregation, i.e. when not all the entities and relationships are mentioned in a single document. Queries for events naturally involve multiple entities, typically the participant in the event and a reference to another item, e.g. *barrack obama birth certificate*.

Clearly, these two categories represent orthogonal aspects, namely the ambiguity of query terms and the complexity of the intended information need. Many queries in the long tail are both ambiguous and complex. They require *deep understanding* of the need and the information behind the query and data, respectively.

## 1.2 Large Amount of Semantic Data

Targeting these problematic cases, there is a category of search solutions – often referred to as semantic search solutions – that aim at exploiting the large and increasing amount of semantic resources that have been made available in different settings. Prominently, Google refers to this as the next generation of search in a recent issue of the Wall Street Journal.

Semantic resources are used by these solutions for the interpretation of queries and data. They include taxonomies, thesauri and formal ontologies. More generally, also structured data exposed as graph-structured RDF<sup>2</sup> data are

1. For other notable persons named George Bush, see [http://en.wikipedia.org/wiki/George\\_Bush\\_%28disambiguation%29](http://en.wikipedia.org/wiki/George_Bush_%28disambiguation%29)

2. RDF stands for Resource Description Framework, a W3C Semantic Web standard for data representation and exchange on the Web: <http://www.w3.org/RDF/>

regarded as semantic resources (often called *semantic data*). They come as standalone RDF datasets or are embedded in Web pages in the form of RDFa (a W3C standard<sup>3</sup> for embedding RDF data in Web pages). These data have been increasing rapidly. As a result of community projects such as Linking Open Data<sup>4</sup>, a large amount of structured datasets formerly kept protected in databases are now exposed as publicly Web-accessible RDF data (e.g. Linked Data<sup>5</sup>). Besides large companies such as BestBuy and Facebook, also national governments such as the US and UK have followed this direction of publishing and linking semantic data on the Web. As a result of active adoption and support from Web search engines providers such as Google and Yahoo!, 10 percent of all Web pages are estimated to contain some form of semantic data markups such as RDFa [6].

The main idea behind semantic search solutions is to use these semantic resources to improve the search performance. Taxonomies, thesauri, and ontologies capture semantics that can be used to understand the query and data and to *resolve ambiguities*. Instead of returning textual data (Web pages, documents), precise facts capturing entity related information and their relationships can be directly retrieved from semantic data to provide *direct answers to complex queries*.

## 1.3 Plethora of Semantic Search Solutions

Capitalizing on the opportunities that arise from this large and increasing amount of semantic resources on the Web, commercial search engines have made different kinds of semantic search features available to end users. The first well-known example towards this direction is Powerset<sup>6</sup>, which makes use of semantic data extracted from Wikipedia to answer complex questions. Semantic data embedded in Web pages are now actively used by Google to provide rich result snippets<sup>7</sup>.

Likewise, research interest in semantic search is stirring up. A wide range of semantic search approaches have been proposed by researchers from different communities, including database, Information Retrieval (IR) and Semantic Web. Correspondingly, semantic search has been viewed from different angles, resulting in a plethora of solutions.

Underlying all these efforts lie one central idea, namely to use semantic resources for more effective search. However, existing approaches greatly vary in the *data*, (2) the *semantic resources*, the *information needs* and (4) the *querying paradigms* that are supported. Most notably, there are principle differences between semantic search approaches, which use semantic resources to improve document retrieval, and the ones that compute direct results, i.e. semantic data retrieval. That is, whereas the one category of approaches focuses on textual data, the other category

3. <http://www.w3.org/TR/xhtml-rdfa-primer/>

4. <http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>

5. <http://linkeddata.org/>

6. [http://en.wikipedia.org/wiki/Powerset\\_\(company\)](http://en.wikipedia.org/wiki/Powerset_(company))

7. <http://www.google.com/webmasters/tools/richsnippets>

targets the case of structured and semantic data. The specific types of semantic resources used by these approaches vary and may include thesauri, full-fledge ontologies or large amounts of semantic data. With respect to information needs, solutions focusing on the retrieval of single entities (and documents) can be distinguished from relational ones, where relationships between entities have to be considered. Substantial differences also exist between systems, which either use NL, keywords or facet-based operations as the primary means for querying. Further, research work found in literatures often studies one of the specific (5) *subproblems in search*: Namely, different concepts and techniques have been proposed for (5a) the interpretation of query inputs and data, (5b) matching the query intent against data and (5c) ranking results.

## 1.4 Contributions

Different solutions for semantic search have been around for a long time. In this work, we present a systematic survey to provide a taxonomy of various semantic search approaches, to reflect on the achievements that have been made over these years, and finally, to discuss the challenges ahead and directions for future research.

There exist a few usability studies, which compare several semantic search solutions [45], [66]. However, this line of work focuses on the differences in querying paradigms. Towards a classification of semantic search approaches, Mangold discusses existing solutions along the dimensions of architecture, coupling, transparency, user context, ontology structure and ontology technology [52]. The last two criteria basically capture the differences in semantic resources among existing systems, which are also discussed in our study. In addition to query paradigms and semantic resources, our study includes the aspects of data, information needs, and provides a fine-grained taxonomy that distinguishes approaches by the subproblems in search they addressing. Further, whereas the previous survey [52] focuses on document retrieval, our study aims at a unified view of document and data retrieval. In particular, we cover a large body of recent work that is not part of that previous survey.

## 1.5 Outline

The paper is organized as follows: An overview of semantic search and the concepts it entails is presented in Section 2. Then, the paper is organized mainly along the aspects based on which we distinguish different semantic search approaches. Different types of information needs, querying paradigms, data and semantic resources are presented in Section 3, 4, 5 and 6. Then approaches are discussed along the subproblems of content interpretation, query interpretation, matching and ranking in Section 7, 8, 9 and 10. After that, we present the main types of semantic search approaches and discuss their performances in Section 11. Main challenges derived from that and implications for future work are presented in Section 12, before we finally conclude in Section 13.

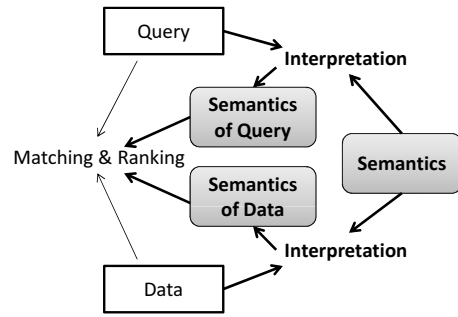


Fig. 1: Semantic Search.

## 2 SEMANTIC SEARCH

Basically, *search* can be characterized by three components, namely the *data*, the *query* and the *matching* framework for computing results from the data for a given query. This is illustrated in the left portion of Fig. 1. One main characteristic of search concepts is its emphasize on end users. That is, search typically involves intuitive interfaces targeting users, who may do not possess knowledge about the domain, the data and structured query languages. The query inputs that can be obtained from these interfaces are often ambiguous. That is, they allow for many possible *interpretations*, which in turn, yield many possible results that vary in relevance w.r.t. the intended query intent. *Ranking* is an essential component in search that aims to address this, i.e. to distinguish results in terms of relevance. Basically, matching can be conceived as the task of identifying matches whereas ranking is more specifically about the degree of matching.

*Semantic search* is a concept widely used by different communities to refer to search approaches that broadly speaking, use semantics to improve the search experience. Considering the core search tasks, semantics has been used to obtain a better understanding of the data and query as well as to improve matching and ranking based on that (see Fig. 1). There are exist different notions of semantics. One the one hand, there are proposals for using the semantics of words captured as latent concepts (e.g. inferred through Latent Semantic Indexing [38]) or latent topics (e.g. inferred through Latent Dirichlet Allocation [76]). While these models only implicitly capture the semantics in terms of some latent, unknown dimensions, there are also explicit models of semantics. For instance, Explicit Semantic Analysis uses Wikipedia articles, categories, and relations between articles to capture semantics in terms concepts, which are then used for concept-based IR [?]. Other explicit models of semantics (henceforth called *semantic models*) include thesauri, the terminological part (TBox) of ontologies and data schemas. While these models capture semantics at the conceptual level, there are also *semantic data* capturing the semantics of individual entities. To accommodate the wide range of semantic search approaches that exist, we formulate the following general notion of semantic search:

*Definition 1: Semantic search* is a search paradigm that makes use of explicit semantics to solve core search tasks,

i.e. to use semantics for interpreting query and data, matching query against data and ranking results.

While all semantic search approaches involve some kinds of explicit semantics, the retrieval contexts and the specific semantic models used to deal with the meaning behind the query intent and data vary. In particular, we identified the following aspects, based on which we will characterize and distinguish existing solutions:

- the type of targeted *information needs*,
- the representation of information resources (*data*),
- the representation of the information need (*query*) or respectively, the underlying method for querying the data (querying paradigms),
- the *semantic models* used to understand and to represent the meaning behind query and data,
- the framework for *matching* queries against data, which also involves *interpreting* the data and query intent as well as *ranking* the results.

For instance, we can look at the differences in the data to arrive at the distinction between document retrieval and data retrieval, two major research directions in search that mark the borders of large communities.

**Document Retrieval:** The IR community targets the document retrieval context where results are retrieved from a collection of documents (primarily textual data). Semantic search of this type is different from standard document retrieval approaches in the use of semantics. The traditional type of semantic models used here are thesauri that are employed to interpret the terms in the textual representation of queries and data. Recently, semantic data such as RDF resource descriptions have also been employed to interpret query and documents in terms of real-world entities and their relations.

**Data Retrieval:** Querying entities and their relations is a core problem in database research. As opposed to the hidden semantics of textual content in documents, information about entities and their relations are explicitly available in this setting – either directly as semantic data in RDF or as some kinds of structured data that can be converted to RDF. Thus, understanding the semantics of the underlying data is not the problem here. However, as opposed to standard data retrieval that is based on structured query languages, intuitive query interfaces are needed in the search context. This leads to the issue of ambiguity. Hence, semantic search approaches of this type have to deal with the resulting problems of interpreting the query intent and ranking the many possible results.

More fine-grained than this general breakdown into document and data retrieval, Fig. 2 illustrates the specific differences in queries, data, semantics, results (corresponding to information needs) and solutions for the subproblems that distinguish existing semantic search approaches. We can see that central to semantic search is the use of semantics (depicted as gray boxes in Fig. 2). In particular, semantic resources represented by lexical models, knowledge models as well as semantic data and metadata are used for the subproblems of understanding raw data (e.g. text) and queries. The resulting semantics of data and queries are

then employed for solving the subproblems of matching and ranking.

### 3 INFORMATION NEEDS

The main motivation for semantic search is to go beyond the relatively frequent but simple queries currently supported by existing Web search solutions to solve the long-tail queries representing rather complex information needs. But also for simple queries, semantic data have proven to be valuable. Instead of showing documents, semantic data are used by commercial Web search engines to create rich snippets for Web pages, or to deliver direct answers, e.g. a specific address for a query mentioning “location” and a named entity such as “Yahoo! Research Lab Barcelona”. We will now discuss several common types of queries and the information needs behind them, from the more simple entity queries to complex relational and analytical queries.

#### 3.1 Entity Search

Most queries on the Web ask for pages representing entities such as companies or people. They are used to obtain entry points, while additional navigation and browsing are still required to satisfy the actual information need. Thus, in the IR context, these queries are also referred to as navigational queries. Recently, semantic data embedded in Web pages have been exploited to address this type of queries. This was pioneered by Yahoo! in their SearchMonkey program, and later adopted by Google to present rich entity snippets<sup>8</sup> as answers to this kind of queries. As an example, Fig. 6 illustrates the entity search functionalities provided by Yahoo!. Not only there are Web pages but also entity descriptions appear as results. These descriptions are directly retrieved from semantic data (in this case, a dataset about music artists and their albums). Instead of searching over a collection of documents, semantic search engines such as Falcons [15] and Sig.ma [70] use semantic data only, and return entity descriptions in RDF as results. Examples for this type of queries include the one asking for “young scholars in Germany” already mentioned before, or the query “Researcher Thanh Tran” depicted in the top left portion of Fig. 5. Results for queries of this type comprise one or several entities.

#### 3.2 Factual Search

While results to the previous type of queries presented to the users are either some entity descriptions or Web pages about entities, the information needs behind this type of search requests require specific information from these descriptions and pages, respectively. Many searches on the Web for instance, aim at finding the ratings of restaurants, or the phone number of a particular person. To continue with our example, users may directly ask for “the birth place of young scholars in Germany”. Direct answers to this type of queries are supported by WolframAlpha<sup>9</sup> and True Knowledge<sup>10</sup>, or the various research prototypes such

8. Google Rich Snippets

9. <http://www.wolframalpha.com/>

10. <http://www.trueknowledge.com/>

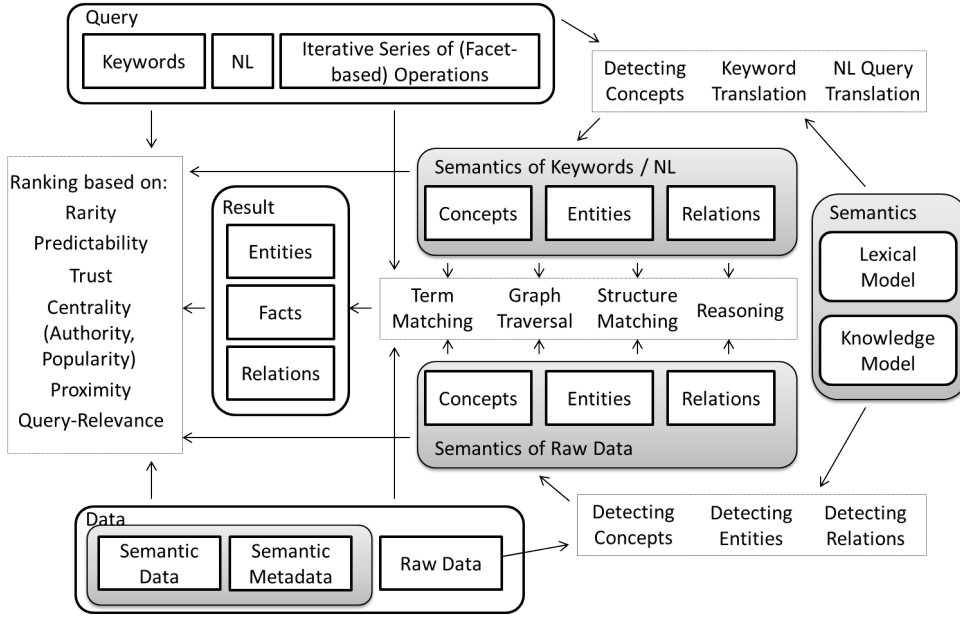


Fig. 2: Overview of Semantic Search Approaches.

as Hermes [67] or AquaLog [51]. Since factual search is mostly about facts that correspond to attribute values of entities, it is often regarded as a particular type of entity search. In this survey, we do not further distinguish this from the general entity search task.

### 3.3 Relational Search

The information need here goes beyond single entities and their factual information. Answers to queries of this type comprise several entities and relationships between them. Hence, processing these queries requires some understanding of relations between entities. For instance, the example presented in the introduction about 32 year old computer scientists requires knowledge about their `lives in` and `birth place` relations to some locations. Relations may be explicitly specified as part of the keyword or NL query, and entities satisfying these relation constraints can be computed using SemSearchPro [65] or AquaLog [51].

Other systems such as NAGA [44] also support searches where relations to be considered are not explicitly mentioned in the query, or simply *unknown*. They address information needs that involve finding how some entities are related to some other entities. The interesting part of the results here are thus not the entities, but the relationships between them – also referred in literatures as semantic associations (direct connection) or property sequences (indirect connections, i.e. path) [3]. An example for this type of query asking for paths between “Peter Mika” and “Thanh Tran” is depicted in the top right portion of Fig. 5.

Even though Yahoo! entity search as shown in Fig. 3 does not directly search for relations, it can be seen on the left portion that relations between entities captured in semantic data constitute valuable information for suggesting related entities. Searching for known relations or unknown

relations is supported by the Information Workbench<sup>11</sup>, a commercial adoption of SemSearchPro. On the bottom part of Fig. [?], we can see results returned to a query that specifically asks for entities connected through the `producer` relation. When relations are not explicitly given in the query but only entities and (or) concepts, this system is also able to explore paths between them. However, this system does not directly return relationships and paths in the data as results, but structured queries representing interpretations of the entered keywords – a querying paradigm we will discuss in the next section.

### 3.4 Analytical Search

Beyond the common searches for entities and relations, which form the focus of this survey, there are also information needs, which can only be addressed when some additional analyses are performed on top of the data. Generally speaking, any kind of additional computation may be applied to the data and results for the purpose of analytical search. Analytics often found in semantic search engines are supported through simple sorting and averaging, more sophisticated statistical and mathematical models, or some form of Machine Learning based inductive reasoning or logic-based deductive reasoning. For instance, WolframAlpha is advertised as a computation engine because it applies different kinds of mathematical models to compute answers to queries, whereas True Knowledge uses an inference engine to perform temporal and geospatial reasoning.

## 4 QUERYING PARADIGMS

Typically, addressing complex needs requires the expertise of technical users who know the underlying data and

11. <http://iwb.fluidops.com>

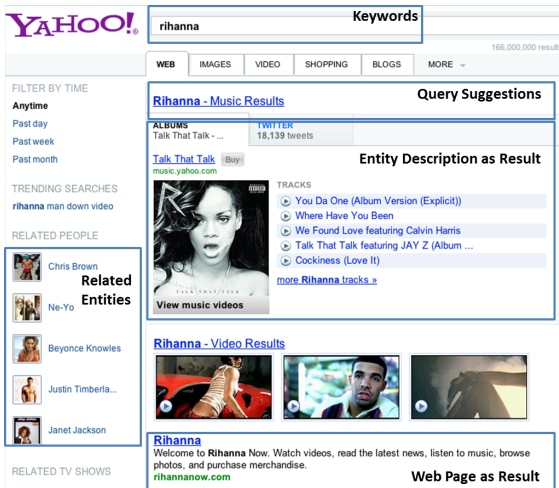


Fig. 3: Entity search with Yahoo!.

schema and use them to formulate questions as structured queries. However, search is commonly seen as an end-user oriented paradigm, which instead of using structured query languages, involves intuitive and easy-to-use access interfaces. The three main interfaces studied for semantic search are based on keywords, NL, and facets. Often, a combination of these paradigms, e.g. keywords and facets, is used to support an iterative search process where instead of constructing the query at once, the user modifies the query and results respectively, through a series of operations.

#### 4.1 Keyword Search

Expressing the information needs as keywords is a paradigm that is not only popular for Web search but also widely adopted by the new breed of semantic search solutions. Most engines focusing on entity search such as Falcons [15] and Sig.ma [70] feature a keyword search interface. Keywords can also be used to formulate more complex relational searches. Addressing this scenario, engines such as Hermes [67], SemSearchPro [65], TASTIER [?] and IBM's AVATAR [42] retrieve entities matching keywords, and search for relations between them in the data.

#### 4.2 NL Search

This is the other paradigm for users to express their needs using their own words. Traditionally, it has been used to pose complex questions against expert systems built for specific domains. Recently, NL interfaces have proven to be also applicable and useful for the multiple domains setting and especially for Web search – as demonstrated by commercial engines such as WolframAlpha, True Knowledge and Apple's Siri<sup>12</sup>.

#### 4.3 Faceted Search

Instead of formulating the entire query at once, *faceted search* supports an iterative process of querying, browsing

and query refinement. This belongs to the broader category of approaches called *exploratory search*, which assumes that users do not precisely know how or what to search and thus, combines querying and browsing strategies to enable learning and investigation. Upon the initial search performed by the user (e.g. using keyword search), faceted search systems [75], [30], [37] present results as well as facets representing attributes and relations that are applicable for refining (or simply, are related to) the current results. These facets can then be used to browse, to refine or to expand the results (i.e. to modify the query, respectively). Microsoft's EntityCube<sup>13</sup> for instance, returns entities upon users' keyword search requests along with relations and attributes of these entity results as facets. Parallax<sup>14</sup> is another commercial faceted search solution (from Freebase, now Google) that enables a combination of search, intelligent result visualization and facet-based browsing.

While most faceted search solutions focus on browsing and refining results of entity queries, faceted search in principle can also be used to address complex needs. For instance, gFacet [37] supports relational search through the construction of complex facet graphs representing different types of entities and relations between them. Fig. 4 shows the faceted search feature from the Information Workbench. Every column in the result table captures one particular type of entities and this system shows facets for every type that appears in the results (e.g. Fig. 4 shows facets for column 1). Hence, entries in every column (representing a partial result and a query part, respectively) can be refined or expanded. For instance, the column of singles shown in Fig. 4 can be further refined to contain only those connected with Freddie Mercury through the *writer* relation.

#### 4.4 Iterative / Exploratory Search

While faceted search is the most popular paradigm in the industry, there are other exploratory search interfaces that support an *iterative search* process. Instead of showing facets as users type, also *completions* of the single user keywords (term completions) or of the entire query representing the full query intent (query completions) have been proposed. One example system supporting this is the Information Workbench shown in Fig. 4, which presents terms matching the current tokens entered by the users, as well as full interpretations of the entire query. For “queen single”, the interpretations shown indicate that the query intent may be something matching the word “queen” that is of the type *single*, or something that is of the type *single*, which is produced (or written) by something else matching the word “queen”. Google, Freebase<sup>15</sup> and True Knowledge also present possible interpretations of the query as the user type or after issuing a query.

Instead query-based completions, also results matching the (possible interpretations of the) keywords provided so far have been presented (called result completions [66]). For

13. <http://entitycube.research.microsoft.com/>

14. <http://www.freebase.com/labs/parallax/>

15. <http://www.freebase.com/docs/suggest>

12. [http://en.wikipedia.org/wiki/Siri\\_\(software\)](http://en.wikipedia.org/wiki/Siri_(software))





and there are links between Wikipedia pages that can be seen as relations. In fact, exploiting this and other specific features of Wikipedia articles such as Infoboxes, a RDF dataset called DBpedia [9] has been automatically derived from this collection. Wikipedia as a source of semantics is particularly popular for search systems that focus on expert search or entity search in general. It is used either to directly answer entity search queries or as background knowledge [43], [11].

**Structured Entity Descriptions:** This type of semantic data is ubiquitous on the Web, capturing information about real-world objects as RDF resource descriptions (or structured entity descriptions that can be directly converted to RDF). Through industry efforts such as Yahoo’s Search-Monkey, Google’s Rich Snippets and Facebook Like<sup>19</sup>, a large amount of RDF descriptions of resources such as restaurants and movies have been embedded into existing Web pages as RDFa. Besides, hundreds of large-scale RDF datasets have been made available on the Web. Started with the Linking Open Data, more and more data are made freely available on the Web and linked with other public datasets. Not only Web content providers but many companies, public institutions and national governments (US, UK etc.) are now actively publishing and linking RDF data on the Web (called Linked Data). Available datasets can be subdivided into broad/domain-independent ones such as as DBpedia and Freebase<sup>20</sup> or deep/domain-specific ones. Large domains covered by Linked Data today include government, health care, entertainment and geography.

**Formal Entity Descriptions:** On the other end of the spectrum, there are more formal views on what constitutes semantic data. In particular, many researchers may argue that the reason why RDF, and other data models that are built upon a formal model of semantics, are actually called semantic data (instead of structured data) is because they can be exploited by a reasoning engine to infer new knowledge (i.e. the knowledge that is entailed by the semantics). While the formal model of semantics specified for RDF is relatively simple (consists only of a few inference rules), semantic data in some existing search systems are represented using more expressive knowledge representation languages such as OWL and F-logic [?] (basically, a combination of logic and object-oriented frame-based languages). For instance, Serene [28] makes use of facts that are captured as Abox assertions (i.e. instance data, as opposed to Tbox, the schema part) of an OWL ontology. ORAKEL [19] searches over an F-logic knowledge base. However, while the use of formal semantics and reasoning has been investigated for some scenarios, it plays only a limited role in existing systems. The majority of approaches does not rely on reasoning. It seems that the research focus still lies in correctly understanding the ambiguous queries and data (namely, to interpret queries and data as elements of the semantic data model as defined above), while reasoning over semantic data may be beneficial but not essential yet.

To sum up, a general notion of semantic data is employed in this work to consider all those systems, which capture the semantics of information resources and real-world objects as entities and relations. Often, these entities and relations are specified on the account of a formal data model with well-defined semantics (e.g. RDF, OWL, F-Logic). However, such a formal model is not strictly required (e.g. in the case of text-rich entity descriptions) and in particular, may not be exploited for reasoning.

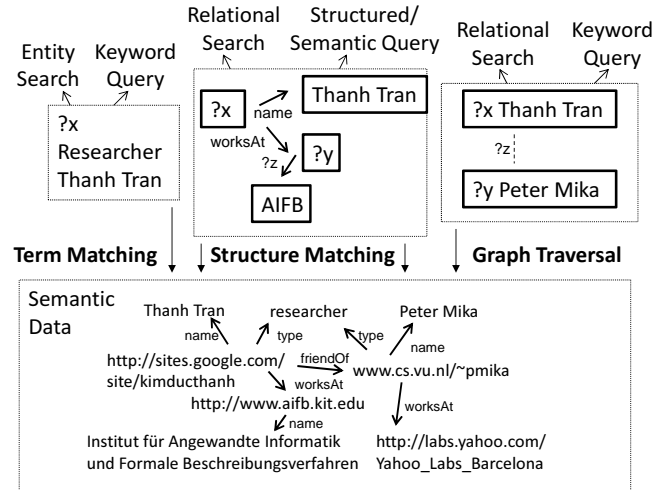


Fig. 5: Examples for semantic data, queries and matching techniques.

There are engines operating directly on semantic data to provide direct answers (data retrieval systems). These are mainly systems, which provide search interfaces over RDF data and ontologies crawled from the Semantic web [15], [67], [39], or structured databases that have been converted to semantic data graphs [49], [50]. Besides them, there are also semantic search systems supporting the retrieval of documents or information objects in general. They employ semantic metadata, often in combination with the raw data representation of these objects.

## 5.2 Semantic Metadata

Basically, semantic metadata is semantic data that captures information about the objects to be retrieved. Because it is clear from the context, these two terms are often used synonymously (in this work). Metadata include basic information about the objects such as author and the year it was published. Moreover, it may capture the semantics of the information contained in these objects. For instance, if a particular Web page is about a restaurant, its metadata may contain a structured description of that restaurant (as RDFa) corresponding to the text contained in the page. In fact, one main challenge in semantic search is to interpret the semantics behind the content of information objects and to represent it as semantic metadata. Not only entities may be recognized but for instance, the content of a document may be interpreted as referring to a meeting with the two researchers Peter Mika and Thanh Tran

19. <http://developers.facebook.com/docs/reference/plugins/like/>

20. <http://www.freebase.com/>



as participants. Semantic metadata representing the content may also be manually produced and embedded into the objects. Recently, several industry projects such as Google Rich Snippets and Yahoo! SearchMonkey actively target this development, providing incentives for site owners to embed RDFa or other kind of semantic metadata into their Web pages. Complementary to this, [schema.org](http://schema.org)<sup>21</sup> is an initiative promoted by major Web search companies (Google, Microsoft, Yahoo!) aiming at the provision of a common vocabulary that can be used by site owners to capture their data. In this regard, we note that while Wikipedia can be conceived as semantic data (with pages corresponding to entities), a more typical viewpoint is to consider it as a collection of documents that are manually associated with metadata: every page is associated with a set of categories and exactly one entity (the entity it is about).

While the amount of available metadata is increasing, it is still too sparse to cover all information needs. Thus, metadata is mostly used as additional information that can help to improve standard search over raw data [14], [29]. Search over raw data can also be seen as a fall-back solution for cases where metadata cannot completely satisfy the information need. There exist also proposals for ontology-based IR, which completely rely on metadata [64]. In this case, the retrieval of documents is formulated as a data retrieval task, where data about the documents (metadata) are returned as direct answers.

## 6 SEMANTICS: SEMANTIC DATA AND SEMANTIC MODELS

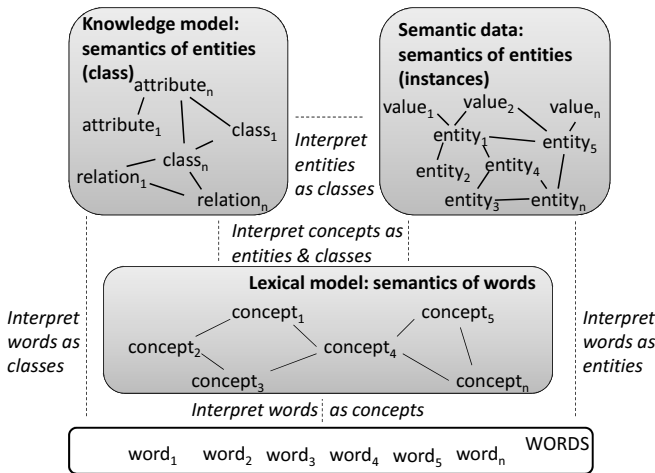


Fig. 6: Words, concepts and entities.

As discussed, central to semantic search system is the use of semantics. The color highlighting (gray) in Fig. 2 indicates that a main source of semantics is the *semantic data* itself. That is, semantic data can be used directly to answer queries (i.e. by matching queries against semantic data), or as a semantic resource to interpret raw data and

queries (as we shall see in more detail in subsequent sections). More specifically, Fig. 6 shows that in the textual raw data context, these semantics can be used to interpret words in terms of entities and relations. It also shows the other sources of semantics called semantic models. Broadly, *lexical models*, which capture semantics at the level of words in terms of concepts, can be distinguished from *knowledge models*, which capture real-world knowledge in terms of entity classes, relations, and attributes.

We have discussed semantic data as one source of semantics in the previous section. In this section, we provide an overview of the semantic models used by current systems, and subsequently, present a notion of semantics that combines the instance-level semantics captured by the data and the schema-level (conceptual) semantics captured by the semantic models.

### 6.1 Knowledge Models

While semantic data capture concrete entities (instances), these are abstract models of knowledge, basically representing classes of entities, attributes associated with these classes, and relations between classes. This is visualized in the top left portion of Fig. 6.

Different models, which vary in the degree of formality and expressiveness have been used by different communities for semantic search. For instance, the model used by ORAKEL for NL question answering is captured by the Tbox portion of an ontology and consists of classes (also called concepts) that are ordered in a subsumption hierarchy and are connected through relations. Range and domain information are captured as restrictions on the relations. As shown for Serene [28], expressions more complex than these restrictions can be added to capture the semantics of concepts to reflect either general knowledge (such as the knowledge encoded in Wikipedia) or specific knowledge of a domain. Because the ontology models used in these systems are represented using the logic-based formalisms F-logic or OWL (description logic, DL), they can be seen as *logical theories* with well-defined formal semantics.

Also, *conceptual graphs* (CGs) have been employed for representing ontology models, and used for the tasks of NL questions interpretation and answering [13] as well as for interpreting and representing documents [22]. CGs combine the intuitiveness of graph-based languages and the formal foundation of logics, enabling the modeling of knowledge in terms of concepts and their relations and its mapping to different formal languages. Corese [23] uses CGs as an internal model (to take advantage of previous work on CGs in the KR community), which however, are then translated to RDFS to conform with this more commonly used standard language for representing RDF schemas.

Closer to the end of linguistic models discussed next are knowledge models that consist of concepts only. For instance, C-Search [32] encodes knowledge as concepts expressed in propositional DL. This DL enables the representation of complex concepts as a conjunction or disjunction of atomic concepts. However, it does not support relations.

21. <http://schema.org/>

## 6.2 Lexical Models

The use of concepts has already been investigated in the early years of IR research [31]. Different to knowledge models, which capture semantics in terms of real-world entities and their relations, the models employed there are lexical in the sense that they capture semantics at the level of words. While concepts in knowledge models stand for classes of real-world entities, concepts in lexical models correspond to *senses of words*. Lexical concepts are more general such that they may refer to entities, classes, relations, attributes, or other senses of words. While these different “types” of senses are not explicitly distinguished like in a knowledge model, concepts can be organized along lexical relationships in a lexical model.

*Thesaurus* is a popular lexical model that has been extensively used for document retrieval in IR, which basically, groups words together according to their semantic similarity. Each group represents a different word sense. In practice, lexical databases also considered as thesauri such as WordNet, or thesauri used in classic IR systems [31], not only capture senses (concepts) and their lexical variants, but also relationships between them such as synonym, broader, narrower and related. This is illustrated in the bottom part of Fig. 6.

## 6.3 Semantics

Based on the above discussion, we introduce a general notion of semantics that can be used to characterize the different types of semantic search systems.

*Definition 3:* The *semantics* of queries and data are captured by semantic search systems in different ways. Semantics can be represented (1) as word senses that are organized through lexical relations between words (*lexical model*), (2) as specific knowledge about concrete entities, i.e. specific attribute values and relations between entities (*semantic data*) or (3) as general knowledge about classes of entities, class attributes, and relations between classes (*knowledge model*).

A prominent example that can be used to illustrate these many aspects of semantics is Wikipedia. As discussed, it has been used as semantic data to answer entity search queries, and converted to a semantic dataset called DBpedia [9]. The Wikipedia categories of pages have been treated as classes and used to construct hierarchies of classes in an ontology called YAGO [62]. Wikipedia is also used as a source for Explicit Semantic Analysis (ESA) [?], where every Wikipedia page is regarded as a concept. Because ESA does not distinguish entities from classes, the concepts here can be seen as capturing word senses. Thus, Wikipedia pages and links constitute a lexical model in this context.

## 6.4 Focus: Semantics of Entities & Relations

The discussion of semantics so far focuses on entities and simple (binary) relations between them. Yet, the kind of semantics that can be expressed is only limited by the knowledge representation (KR) languages. There exist very powerful KR formalisms and many of them have been

incorporated to perform document retrieval (already in the early years of IR [71]) and to represent complex queries and knowledge used in NL question answering [72] (in the early years of expert systems). For instance, the employed formalisms have been used to represent temporal or fuzzy aspects of world knowledge. However, there are practical reasons why the focus of current research (and this paper) is set on simple descriptions of entities (via attribute and binary relations). On the one hand, most common queries as well as a large part of queries in the long tail of Web query logs are about entities and relations (i.e. they are of the types entity, factual or relational queries). More importantly, the bottleneck is actually the availability of data. It became evident that manually capturing the content of documents as complex logical formulas does not scale. Extracting entities and relations from documents at high quality still remain hard problems, representing the limits of what can be done automatically by data extraction technologies. Also, most of the data residing in databases nowadays are mainly about entities and their relations.

# 7 CONTENT INTERPRETATION

One crucial task in supporting semantic search over documents is to obtain a richer understanding and representation of the document collection. Instead of a the typical bag-of-words model used in classic IR, the goal is to capture the content of the documents as entities, concepts and relations, i.e. as semantic data or elements of a semantic model. In fact, there is a dedicated area of research focusing on this problem of knowledge extraction from text. Problems that have been studied include named entity recognition, semantic role labeling and relation extraction. For an overview of solutions proposed in this area, we refer the interested readers to surveys specifically focused on this problem [?]. Here, we discuss the kind of knowledge extraction that has been performed for the purpose of semantic search.

## 7.1 Detecting Concepts in Text

This problem and the tasks carried out to solve it are illustrated in Fig. 7a. Concepts are especially needed for supporting what is known as concept-based IR.

Documents have been analyzed to extract concepts for the task of modeling documents as conceptual graphs [22]. For C-Search [32], which represent documents as DL concepts corresponding to WordNet senses, the authors identify words in documents matching Wordnet senses (i.e. direct matching), and uses part-of-speech (POS) tagging information and Wordnet, as lexical database of senses and word-level relationships, to perform word sense disambiguation. While words are mapped to atomic concepts, phrases are represented as complex concepts using DL formulas [32]. Especially in the biomedical domain, the use of thesauri concepts for interpreting and representing documents is very common [69], [78]. For instance, the Unified Medical Language System and a gene-thesaurus have been used to identify multi-word terms and to map synonymous words

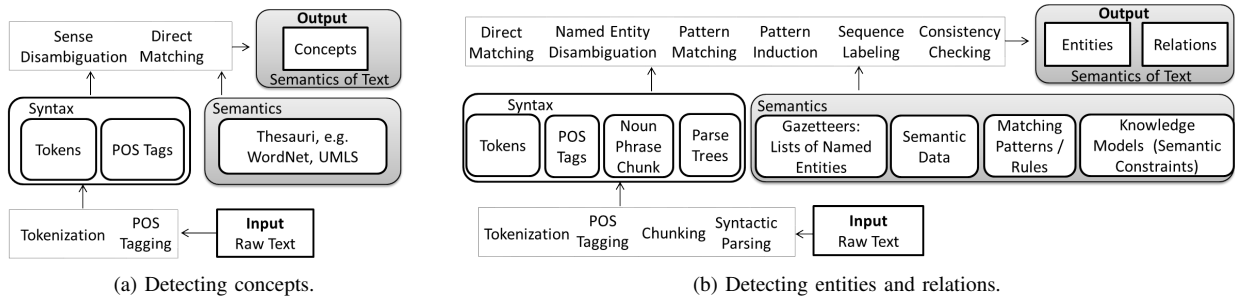


Fig. 7: Detecting concepts, entities and relations.

to one concept [69]. The concepts were identified simply by matching document words to thesaurus terms.

## 7.2 Detecting Entities & Relations in Text

For extracting entities and relations, the large body of research work on named entity recognition and relation extraction has been leveraged by existing semantic search solutions. One of the platforms, which is frequently used by semantic search systems for extracting semantic data and automatically annotating document collections at a large scale is KIM [?]. IBM’s AVATAR [42] makes use of UIMA, a framework of annotators that can be configured in a pipeline to analyze and extract data from text. There is another work from IBM, which focuses on extracting named entities and relations from text and using them for more precise search (employing XML fragments as queries [18], [17]).

An overview of tasks involved in detecting entities and relations are shown in Fig. 7b. As for most text interpretation tasks, a syntactic analysis of the text is needed. In this case, the typical analysis includes tokenization, POS tagging, chunking sentences into noun phrases, and some data extraction systems even derive entire parse trees from the text. Besides the syntactic information derived from text analysis, different semantic resources are used. In particular, Gazetteers are frequently used, which can be seen as a lexical model. The simplest way for detecting entities is simply to match words (tokens) against Gazetteer’s entries representing entities of different classes. For disambiguating named entities mentions in text, it has been shown that semantic data about entities and their relations provides valuable information [46]. The technique behind this is based on graph traversal, which starts from nodes matching the entity mentions. Controlled through spreading activation, paths from these nodes are then traversed to explore the semantic context of these nodes (i.e. entities). These contextual interpretations are then used for disambiguation. This resembles the general technique called collective matching, which applied to this case, assumes entity mentions (words) to form a context. We have a higher confidence that a word matches an entity  $e$ , when other words in this context, match entities related to  $e$ , i.e. the context formed by  $e$  and its related entities corresponds to the word context.

Beyond direct matching and disambiguation, state-of-the-art techniques employ complex matching rules (or matching patterns in general), especially for the more complex task of relation extraction. Given a pattern like *Mr. ? works for ?*, matching texts can be identified and relevant entities and relations can be extracted. Matching patterns are specified manually or learned automatically through machine learning techniques for pattern induction. Current state-of-the-art data extraction programs in fact, not only use simple word patterns, but complex combination of features derived from the words (e.g. orthographic features such as whether the word contain digits or hyphens, word types such as lowercased and capitalized and syntactic features such as POS information), from the context (i.e. words in a text window along with their tags, labels etc.) as well as from background knowledge (e.g. from Gazetteers). These features along with labeled training examples are then used to train a classifier (one for every entity / relation type) or a probabilistic sequence model such as the Conditional Random Field [?]. Besides these learning methods proposed for targeted information extraction, which assume a fixed set of predefined types of entities / relations to be extracted, there are also proposals for open information extraction [?]. So far, semantic search systems focus on the use of targeted information extraction tools (e.g. IBM’s UIMA, SOFIE [?]). While they provide lower coverage, the quality of extracted outputs is higher compared to open extraction tools, an aspect shown to be very influential in improving the precision of semantic search results [18], [17]. Promising is the combined usage of targeted and open information extraction, a direction currently pursuit by Microsoft’s EntityCube.

Many patterns and particularly rules used for extraction explicitly capture when candidates can be considered as matches, i.e. when they are considered as belonging to some types of entities or relations. These patterns can be seen as sources of semantics that feed as inputs to the extraction systems. Besides these matching-specific knowledge, also general as well as domain-specific knowledge stored in ontologies have been used as semantic constraints to filter matches, i.e. those that lead to semantic inconsistencies. For instance, SOFIE [?] prunes extraction outputs that do not satisfy the semantic constraints captured by YAGO. As an example, having the pattern *?x is mayor of ?y* and the knowledge that

`x, mayorOf, y` requires `x, type, Person` and `y, type, City`, we can filter out those extracted `is mayor of` relations that do not involve a person or a city.

## 8 QUERY INTERPRETATION

Essentially, interpreting queries is similar to the task of understanding text in documents. However, these problems are studied from different viewpoints by overlapping but not same communities of researchers. Thus, the emphases are put on different subproblems. Further, besides NL text, also keywords may be provided as inputs to query interpretation. Keywords are different from NL text in the amount of syntactic features that can be obtained.

### 8.1 Detecting Concepts in Keyword Queries

One of the main challenge in dealing with keywords in queries (and words in NL text) is their semantic ambiguity. The same meaning can be expressed in different ways. In particular, there are lexical variants that refer to the same concept, relation or entity (synonymy). One and the same word however, can also have different meanings (polysemy). These are only two prominent problems, and there exist many other subtle ambiguities (e.g. terms overlap or are related in meaning).

Compared to detecting concepts in text as illustrated in Fig. 7a, solutions to the problem of detecting concepts in queries are different in the usage of features: Keywords are also matched and disambiguated against a thesaurus. However, there is no syntactic information such as POS tags that can be exploited.

The use of thesauri to deal with semantic ambiguities of words has long history and has been investigated as the problem of concept-based IR [31]. Concept-based IR is especially needed in the biomedical domain due to the frequent presence of acronyms, homonyms and synonyms. For detecting the underlying concepts, many lexical resources that are abundantly available for this domain, such as the Online Dictionary of Abbreviations from MEDLINE, the MESH thesaurus and the Gene Ontology have been used [77], [40]. The detected concepts are used to expand the query [58], [40]. Concept-based query expansion [58] for instance, detects the concept behind the query, and performs selection and weighting of additional search terms directly based on this concept (instead of the query terms). This and other works in this area, were based on the use of thesauri that capture concepts and relationships. WordNet for instance, was used to represent concepts by WordNet synonym sets, and for expanding query terms by following WordNet links [73]. WordNet links (is-a in particular) were also used to disambiguate query terms and to choose their senses [74].

Recently, a technique based on the idea of relevance model has been proposed to recognize the concepts underlying the query, referred to as conceptual query modeling [53]. Instead of detecting the concepts based on a thesaurus, this approach retrieves the top pseudo-relevance

feedback documents through an initial query run. Concept annotations associated with these documents are then used to construct the conceptual representation of the relevance model (the query). While OntoSearch [41] does not explicitly target the approximation of a relevance model, it follows a similar idea. Concepts captured by ontologies (seen as a semantic network of concepts) are used here. First, OntoSearch obtains a set of documents for the given query, and then uses the concepts associated with these documents as seeds to the semantic network. Through spreading activation, concepts that are semantically related to this seeds are inferred, and used to re-rank the documents.

### 8.2 Translating Keyword Queries

There are also proposals for mapping keyword queries to fully structured queries. In this case, not only concepts but also the entities and relations to which the query keywords may refer to, have to be recognized. The resulting semantics of query keywords, i.e. schema (concept and relation names) and data elements (entities) provides the basis for identifying the query intents, i.e. the keyword query interpretations. The assumption behind many approaches for query interpretation is that the intended query corresponds to a particular substructure in the semantic data or schema graph. However, relations may be not explicitly specified in the query, or are not sufficient to connect the detected entities and concepts. In this case, algorithms for graph traversal are used to find connecting paths. Query interpretations are then obtained by merging these paths, which are finally ranked and mapped to structured queries (i.e. concept, relation and entity names are mapped to query predicates and constants). While the common language used to query semantic data in RDF is SPARQL<sup>22</sup> (basically, a language for expressing graph patterns), other (logic-based) formalisms have been used, i.e. recognized entities, classes and relations can be mapped to SPARQL queries [24], conceptual graphs [13], logical formulas [19] or XML Fragments [18]. An overview of this task is illustrated in Fig. 8a

AVATAR [42] is an example of systems supporting this kind of keyword query interpretation. It maintains a translation index, which can be conceived as a lexicon that returns all meanings for an individual keyword. Specifically, it returns schema concepts as well as schema paths matching the keywords. These keyword matches are then combined to enumerate all possible interpretations of the query. For RDF data, a top-k graph traversal algorithm has been proposed to efficiently compute all possible interpretations [68]. Here, keywords are interpreted as entities, classes or relations (called keyword matching elements), which are contained in the semantic data or semantic model. All possible paths between these keyword matching elements are computed online by traversing a query search space that is formed by combining the semantic data graph and the schema graph, and then merged to obtain interpretations, which cover

22. <http://www.w3.org/TR/rdf-sparql-query/>

all the query keywords. This keyword query interpretation is implemented by Hermes [67], and later, SemSearch-Pro [65]. TASTIER [49] also applies a graph traversal algorithm. However, it does not compute possible queries (i.e. interpretations) but directly output answers. As the user types, TASTIER completes the query with answers that possibly match the intended information need.

### 8.3 Translating NL Queries

There are two main categories of approaches for dealing with this problem. On the one hand, it can be tackled in a way similar to the translation of keyword queries. This solution (depicted in the bottom left portion of Fig. 8b) involves the detection of concepts, entities, and relations [13], and their mapping to elements of structured queries. The problem of ambiguity is solved here by using robust techniques for the matching of words in the NL query against elements in the semantic data and semantic models, and for the ranking and disambiguation of resulting matches. In fact, this kind of approaches is particularly popular among domain-independent NL query translation systems. AquaLog [51] for instance, maps words in NL queries to elements in the semantic data, and disambiguates results using domain-independent lexical resources (e.g. WordNet).

As depicted in Fig. 8b, the other category of approaches involves the use of lexicons, often in combination with a syntactic analysis tools. In fact, the same text analysis tools used for content interpretation can be applied here. For instance, the Stanford Parser <sup>23</sup> and GATE <sup>24</sup> are used by FREYA [24] to identify concepts in NL questions and to derive the syntactic parse tree, respectively. The use of domain-specific lexicons provide is a common way to deal with ambiguities, especially for traditional NL translation systems that are focused on a particular domain. Basically, a lexicon captures the mappings from syntactic elements to semantic elements. For instance, ORAKEL uses a lexicon, which specifies the mappings between nouns (verbs) with their arguments identified through the syntactic analysis and concepts (relations) in the semantic models [19]. These mappings are specified by lexicon engineers. A different lexicon is needed when porting the NL interface from one domain to one other. Aiming at reducing this upfront customization effort and domain independence, there are also systems that in the case of lexical ambiguities, ask users to provide mappings, and use them to train models (e.g. via reinforcement learning [24]) that automatically compute mappings.

### 8.4 Iterative Query Construction

The discussed query interpretation approaches aim at a more precise understanding of the information need, represented as a structured query. This is also the objective of other related approaches that fall into the general category

of query construction. For instance, faceted search [75], [37], [30] as well as the other kind of iterative query interfaces based on visual manipulation (drag and drop) [34] as presented before, also belong to this general category. In these systems, user operations are directly mapped to changes in the query. For instance, adding and removing facets result in the addition (query expansion) and removal (query refinement) of query predicates, respectively. Clearly, the difference to work on keyword and NL query interpretation is the lack of ambiguity, i.e. the semantics of every user input is clear.

## 9 MATCHING

TABLE 1: Overview of matching approaches.

Term	Structure	Semantic
Syntactic-based	Join	Forward chaining
Lexicon-based	Graph traversal	Backward chaining

(**Thanh: some systems may do interpretation, some not....**) The differences between term matching, structure matching and graph traversal are illustrated in Fig. 5.

### 9.1 Term Matching

This matching is needed when the representation of query and data is based on terms (i.e. is a bag of words). This is the case with all the existing concept-based document retrieval solutions [31], [77], [40], [58], [58], [73], [74], [53], which identify the concepts behind the query and content but however, only use them for query or document expansion (i.e. expand the term-based query / content representation with additional terms corresponding to the identified concepts). Likewise, there are approaches, which interpret the query as entities and relations and to use this understanding to expand the queries with corresponding terms [54]. Many semantic data retrieval systems such as Falcons [15] and Sig.ma [70] also employ bag-of-words representations – both of the query and the semantic data. In these cases, the structure and semantics are not taken into account during the matching (but possibly, might have been used in other steps such as query interpretation). The matching boils down to the standard IR retrieval problem. Namely, given a set of keywords, documents matching individual keywords are retrieved. Typically, they are then joined to obtain those, which match all query keywords (AND-semantics).

### 9.2 Matching Complete Structure Patterns (Structure Matching)

This matching is possible when the query and content are interpreted or are directly available as a structured query and semantic data, respectively. That is, the semantics of both the query and data is represented in terms of some entities and their relations. Most frequent is the combination SPARQL queries and RDF data. In particular, the three

<sup>23</sup>. <http://nlp.stanford.edu/software/lex-parser.shtml>

<sup>24</sup>. <http://gate.ac.uk/>

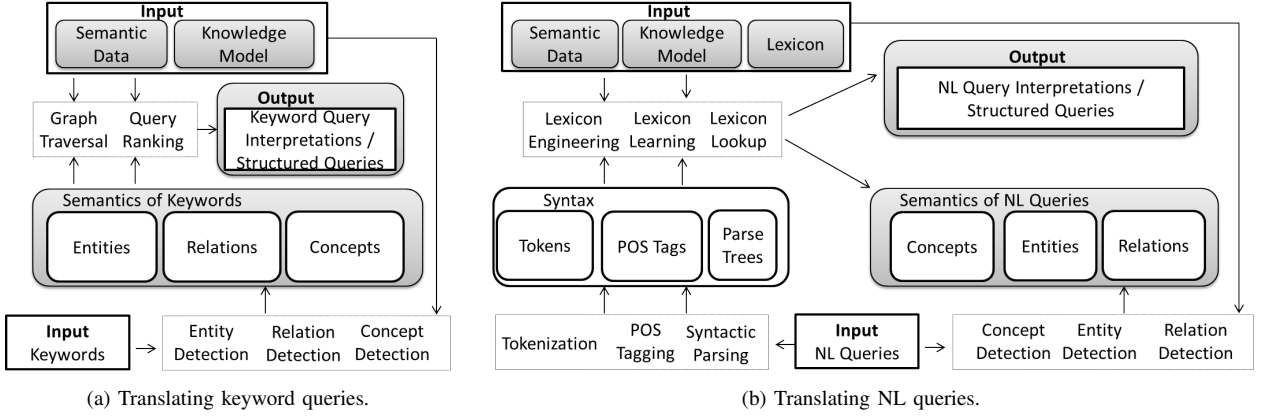


Fig. 8: Translating keywords and NL queries.

common types of information needs discussed previously can be captured through the basic graph pattern (BGP) feature of SPARQL. Just like semantic data, a BGP forms a graph where nodes and edges are either constants or variables. They form complete patterns in the sense that through explicitly captured join variables, the query engines know how to combine intermediate results obtained for subparts of the queries (for triple patterns in the BGP) to compute end results. Matching BGPs against semantic data boils down to the task of *graph pattern matching* where constants in the BGP representing entities, relations, attributes and attribute values are matched against the RDF data graph to find matching subgraphs that contain bindings to variables in the BGP. Processing these BGPs involves standard *database query evaluation* techniques, i.e. precompile the query to capture an optimal evaluation plan, then *retrieve* data (from optimized indexes) for each triple pattern and *join* intermediate results according to the plan. This matching is supported by off-the-shelf triple stores. For instance, it is used by SemSearchPro [65] for computing results for SPARQL queries, after they have been derived from keywords through query interpretation.

### 9.3 Matching Incomplete Structure Patterns (Graph Traversal)

While join-based evaluation is possible for structured pattern that are complete, traversing the data graph is needed when query patterns entail missing connections. That is, the query engines cannot directly combine partial results based on join variables but needs to explore for different paths between them. One combination that involves this type of matching is the evaluation of keyword queries over semantic data. As discussed before, the structure and semantics in the data can be used to interpret the keyword query and to convert it to a structured query. Instead of query interpretation, there are also systems [47], [50], which use the same *graph traversal* mechanism to directly evaluate the keyword query. They skip the computation of queries and directly explore for results in the data (subgraphs) that match the keywords. As opposed to standard IR-style keyword search on documents, the results here are

semantic data graphs. That is, the keywords in the query do not refer to single documents but possibly, several entities that are connected through paths in the semantic data graph. Hence, simply joining results (documents), which match the query keywords is not sufficient. Entities matching keyword(s) in the query have to be retrieved, and subgraphs containing paths, which connect these keyword matching entities have to be explored in the data through traversing graph nodes and edges.

A different semantics has been used [59], where results do not have to cover those nodes matching the query keywords. Those parts of the data, which are interesting and thus, shall be explored and returned as results, are controlled through manually defined traversal constraints that are realized via spreading activation.

Interestingly, graph traversal is also employed for the relational search scenario, where the relations (paths) between entities are unknown. For instance, the mechanism used by for graph exploration by Hermes [67], [68], a keyword search system, is similar to the one proposed for answering P-Queries [3], a special type of queries that ask for unknown semantic associations between entities.

Another application of graph traversal has been proposed for the combined scenario of document and data retrieval. The goal is to find entities in the knowledge base, which are related to the documents to be retrieved, or vice versa, to retrieve documents related to some given entities. The starting points here are thus documents or entities, and traversal from these points are controlled through spreading activation [60].

### 9.4 Reasoning

The formal well-defined semantics of some semantics models such as ontologies represented in OWL, have also be exploited for matching. More precisely, they are used for *reasoning* to consider not only the available data but also new facts that can be automatically inferred from the formal semantics. For instance, the semantic models used in Serene [28] is an ontology TBox represented as DL formulas, and semantic data are captured as ABox assertions of the same ontology. Serene uses an inference

engine to derive new knowledge during an offline ontology compilation step. This computation is performed offline to improve the efficiency of online matching. Clearly, new knowledge relevant for the query can also be derived online. In ORAKEL for instance, the inference of new facts is performed as part of the computation of query matches. Notably, the first system created in the early years of Semantic Web research, which makes use of logical reasoning for computing semantic matches is SHOE [36].

## 9.5 Matching: Combination of Techniques

## 10 RANKING

Ranking approaches leverage different factors, which in general, can be *query-independent* or *query-dependent*. The latter considers the *relevance* of the result with respect to the query. The former includes different aspects, such as *rarity* (measured in terms of occurrence [1]), *popularity* (based on occurrences [68], or centrality [1]), *predictability* (based on information content [2]), *trust* (e.g. trust assigned to sources [1]) or *confidence* (e.g. confidence scores of data extraction outputs [56], [16]). They could be seen as heuristics, which can complement the ranking based on query-relevance. Among those heuristics, there are two prominent ones that are widely used in existing approaches, namely *centrality-* and *proximity-based ranking*.

TABLE 2: Overview of centrality-based ranking.

Hyperlinks	Relations
Adoption of PageRank to (1) ontologies as documents, (2) two-layered graph with sources (documents) representing one layer	Authority transfer weights (1) specified by experts, (2) learned through simulated annealing or PARAFAC decomposition

### 10.1 Centrality-based Ranking

The algorithms commonly used for computing centrality in the context of ranking is PageRank and HITS. Basically, the idea behind it is to perform link analysis on the graph formed by the hyperlink structure of documents to identify nodes that are important (or popular or authoritative). In the semantic search context, much effort has been invested in extending the original PageRank and HITS proposals towards dealing with semantic data graphs, where instead of documents and hyperlinks, there are entities that are connected through different types of semantic links. One of the first proposal for using PageRank for dealing with semantic data is OntoRank [26]. However, ranking here is only dealt with at the level of sources, namely ontologies (instead of the entities contained in them). PageRank could be directly applied because ontologies are treated as Web pages, and there are no different types of links. ObjectRank [4] build upon PageRank, but recognizes the variety of semantic links. Instead of using the same authority flow for all edges, it employs an authority transfer schema graph, which captures the strengths of authority for different types

of edges. While all these weights has to be manually specified by domain experts in ObjectRank, PopRank [56] incorporates a simulated annealing algorithm to learn these weights, based on a partial weighting provided by experts. A different direction for obtaining the weights of semantic relations is taken by TripleRank, which models data by means of a 3-dimensional tensor. Applying the PARAFAC decomposition (a multi-modal counterpart to HITS) then results in authority and hub scores for both the entities and links. EntityAuthority [61] operates not only on the semantic data graph but on a combined graph that contains both entities and documents as nodes. Also, a two-layer approach has been pursuit, where PageRank is computed both for the sources and the entities contained in them [25], [35].

### 10.2 Proximity-based Ranking

Another popular heuristic that has been successfully used in IR is the proximity between terms [12], [63]. According to this, a result is ranked high when it contains query terms that appear closer to each other in its textual content. Note that while it is not independent of the query, this heuristic does not directly capture the query-relevance. This notion of proximity has also been used for XML data retrieval: XRank proposes to rank results based on the smallest text window, which contains all matches to query keywords [33]. The notion of proximity has also been adopted to the case of semantic data, where instead of textual proximity (distance between terms), the distance between semantic entities is taken into account. For document retrieval, several proximity measures have been proposed to measure the distance between named entities extracted from text [48]. In keyword search over semantic data, results are subgraphs containing nodes matching the query keywords. One of the factor often used for ranking these results is based on the (pair-wise) distances between keyword matching nodes (the length of the paths connecting them) [47], [68], [49]. Similar to text proximity, the intuition here is that more compact results more likely correspond to the intended need.

TABLE 3: Overview of query-relevance based ranking.

Content	Structure
Treat entities as documents and adopt (1) Vector Space Model (2) unstructured LM	(1) BM25F, (2) structured LM, (3) edge-specific relevance model, (4) triple-based LM

### 10.3 Query-Relevance Based Ranking

Given the ambiguous query, different results can be found that vary in the degree relevance. This classic problem is extensively studied in IR. In the semantic search setting, existing IR concepts have been used and extended to deal with semantic data and to leverage the availability of semantics.

Many semantic search systems, which provide keyword search over semantic data, apply standard IR ranking. For



instance, Falcons [15] uses a standard IR tool (Lucene<sup>25</sup>) to index entities as documents, and uses the built-in ranking mechanism for entity ranking.

For ranking more complex (relational search) results, there are keyword search systems, which use the Vector Space Model and adopted the *TFIDF-based* term weighting. The result in this setting is a graph, which contain a set of keyword matching nodes. A popular ranking is based on sum of the individual TF-IDF score computed for every keyword match via pivoted normalization weighting. Specific normalization methods have proposed to recognize that the textual length of the matching elements is often very short (document length normalization), different attributes can be associated with different vocabularies (IDF normalization) and a result is actually an aggregation of documents and thus requires additional normalization beyond the document level. Also TF-IDF has been adopted for the case of document retrieval using semantic data. Here the TF-IDF weighting has to be modified to consider not only the frequency of terms but also the frequency of named entities [54] and annotations [14].

Taking the specific semantics of entity results into account, an extension of the *BM25F* has been proposed [10]. The idea is to use different fields for indexing different properties of RDF entities. Different weights are assigned to these fields to recognize that some properties are more important than others in ranking entities.

Also, the more recent *language modeling* (LM) paradigm has been adapted to this case of complex results. Language models for documents and queries are employed to represent them as multinomial distributions over words of the vocabulary. The KL-divergence, basically measuring the distance between the query and document distribution, can then be used for ranking. For ranking structured objects, three different models were studied [55]: The simple unstructured model treats all object attributes and values as vocabulary terms. The structured variant employs language models for specific components of the structured results. The record-level representation models an object through several language models, each capturing a record of that object. The attribute-level representation further segments a record into multiple fields and assigns different weights to individual fields (similar to the idea behind *BM25F*). Moreover, specific term distributions for fields have been proposed. Using edge-specific relevance models [8], the probability of observing a word in a structured result may vary, depending on the attribute (the edge) to be considered. This edge-specific model is employed not only to exploit the structure and semantics in the data but also to capture the semantics behind the keyword query. For this, an initial run over the semantic data is performed to obtain pseudo-relevance feedback (PRF) results for the keyword query. Instead of using the short and ambiguous keyword query, an edge-specific model constructed from these PRF results is used to capture the information need (the relevance).

As opposed to the approaches mentioned before, which

model queries and documents based on words, there are also language models defined as probability distributions over RDF triples [27]. The probability of a given triple should capture its “informativeness”, which is measured based on witness counts. The authors implement this by issuing keyword queries for each triple using Web search engines, and used the reported result sizes as witness count estimates. Following the same direction, NAGA [44] not only uses informativeness but also a measure of confidence to estimate parameters of the language models.

## 10.4 Ranking: Combination of Factors

Clearly, the ranking implemented by a system is often a combination of factors. For instance, for searching entities detected from documents, EntityRank [16] rank entities based on the popularity of documents they have been extracted from (computed via PageRank), the confidence of the extraction outputs, contextual constraints as well as the proximity among entities and query words. For retrieving more complex objects (comprises multiple records) extracted from web documents, experiments with Libra (a scientific Web search engine) have shown that ranking can be improved when taking extraction confidences both at the record- and attribute-level into account [55]. Structured results (a subgraph) obtained from keyword search over structured data is mostly ranked based on the PageRank of its constituent nodes, aggregation of query-relevance scores obtained for individual nodes (records) matching query keywords, and the proximity defined through the length of paths between nodes [68].

(Thanh: todo:learning to rank [21])

## 11 MAIN APPROACHES AND THEIR PERFORMANCES

Semantic search has been investigated from different communities in different context. In the IR community, semantic search has mainly been viewed from the perspective of *document retrieval*. Here, classic *conceptual* approaches based on the use of lexical models can be distinguished from more recent approaches that employ of *annotations in the form of semantic data*. Another line of research is *entity search*, which has been studied in the context of entities extracted from Web pages, Wikipedia pages corresponding to entities, and entities captured by databases and Semantic Web datasets. Another popular topic is *keyword search over databases and Semantic Web data*, which is mainly about relational information. Last but not least, the *processing of NL questions* entail specific challenges and opportunities, which has attracted attention from a specific community of researchers. We will now discuss the role of semantics in solving these different search tasks and its impact on performance.

### 11.1 Conceptual Document Retrieval

This is the classic type of semantic search systems, which has been studied already in the early years of IR

25. <http://lucene.apache.org/java/docs/index.html>

research. Researchers have recognized for a richer representation of the information needs and data that goes beyond the bag-of-words model. It became evident that constructing a full-fledge logic-based representation [71] for queries and especially for documents is not practical. Thus, lightweight *lexical models* have been employed to interpret the semantics of query and documents as concepts, and used within standard bag-of-words IR paradigm (e.g. vector space model and language model). Then, retrieval may be done based on concepts alone, or in the form of query and document expansion. In the latter case, bag-of-words model constitutes the basis while the conceptual understanding helps to refine or add more relevant words. However, there are *no clear evidences suggesting conceptual IR can outperform standard IR techniques*. It has been shown that automatic query expansion in the large diverse TREC collection using WordNet degrades performance. Even when selected by hand, it helps only when the query is an incomplete description of the need [74]. More recently, it has been shown for the medical domain (TREC Genomics) that retrieval based on concepts alone could not outperform the standard baseline based on words [69], [78]. Current work in this direction, which combines concepts and bag-of-words in the form of query expansion, outperforms a state of the art baseline in one collection, and shows similar performances in other collections [53]. However, the experiments have been performed on collections where documents have been manually annotated with concepts (domain-specific CLEF and TREC Genomics collections).

## 11.2 Annotation-based Document Retrieval

This encompasses the newer type of approaches, which exploit advancements in data extraction technologies to obtain a richer representation of queries and documents, namely as entities and relations represented as annotations (query and document annotations, respectively). To date, there exist only anecdotal evidences for the merits of this type of systems. Possibly, the most conclusive result in this direction is based on experiments using the TREC Question Answering track: For a subset of manually chosen questions, it has been shown that higher precision could be obtained [18]. The results there can also be interpreted as follows: The employed data extraction programs were optimized for a few specific types of relations and entities (e.g. Person). High precision is possible for questions that can be covered by these types of annotations. However, high recall is difficult because it requires the data extraction program to recognize arbitrary types of entities and relations. While there are promising ideas towards this kind of open data extraction, this problem remains an open challenge. Along this line, there is a study showing that the quality of data extractions significantly impacts search performance [17].

## 11.3 Entity Search

While the two mentioned types focus on documents, this category includes search over (1) documents (for processing

factoid and list questions of the TREC QA track [18], or for expert search questions of the TREC Enterprise track [5]), over Wikipedia data, which as discussed, can be seen as a kind of semantic data, or (2) documents associated with metadata, (for answering entity related questions of the INEX Entity Ranking track [43], [57]), or over pure (3) semantic data (for answering entity queries of the SemSearch Entity track [10]). Some type(1) systems can also be categorized as annotation-based document retrieval systems because they detect entities and relations in data and queries [16], [55], [18]. As discussed, it has been shown that precision can be improved when high quality extraction outputs are available [18]. The use of semantics in type(2) systems is limited to viewing Wikipedia pages as entities [43], exploiting categories associated with them [6] and links between them [57]. It has been shown that Wikipedia articles can be effectively used as a pivot for entity search on the Web: it contain entities for a large amount of queries as well as cues (e.g. external links) that can be used to effectively find entities' official homepages on the Web [43]. As a source of semantics, experiments have shown that category information helps to interpret the query and entity documents and to outperform standard text-based retrieval [43], [6]. While many type(3) systems [15], [39], [70], [67] have been built to search for entities over data crawled from the Semantic Web, solutions for ranking and the evaluation of their results have attracted interest only recently. The best performed method [10] is based on BM25F, which ranks entities as documents that are segmented into different fields corresponding to entity attributes.

## 11.4 Relational Keyword Search

This category comprises all approaches, which process keyword queries over semantic data [68], [47], [50]. While the results here include entities, the focus is to find possibly complex subgraphs encompassing several entities and relations between them (i.e. to support relational search). There exist a variety of indexing strategies and traversal algorithms that help to perform this task more efficiently. However, recent work [20] has shown that the proposed ranking strategies ?? based on the adoption of proximity, TFIDF and PageRank do not perform well when considering a broad range of queries. This work aims at a standardized framework and benchmark for evaluating the effectiveness of relational keyword search approaches [20]. Based on this benchmark, it has been shown that best performance can be achieved through the use of edge-specific language models that are constructed for results and queries (based on PRF results) [8]. (**Thanh: compare to recent learning to rank approach [21]**).

## 11.5 NL Question Answering

(**Thanh: are there benchmarks for NL QA over data?**)

# 12 MAIN CHALLENGES

- Open data extraction vs. ranking methods on low quality extractions?

TABLE 4: Add caption

		C-Search	IBM	Falcons	Yahoo!	AquaLog	Hermes	Naga	TASTIER
Need	Entity								
	Fact								
	Relation								
Query	Keyword								
	NL								
	Facet								
	Iterative								
Data	Raw Data								
	Semantic Data								
	Semantic Metadata								
Semantic Model	Lexical								
	Knowledge								
Interpretation	Concept								
	Entity								
	Relation								
Matching	Term								
	Structured Pattern								
	Reasoning								
	Traversal								
Ranking	Centrality								
	Proximity								
	Query Relevance								

## 13 CONCLUSION

## REFERENCES

- [1] B. Aleman-Meza, C. Halaschek-Wiener, I. B. Arpinar, C. Ramakrishnan, A. P. Sheth, Ranking complex relationships on the semantic web, *IEEE Internet Computing* 9 (3) (2005) 37–44.
- [2] K. Anyanwu, A. Maduko, A. P. Sheth, Semrank: ranking complex relationship search results on the semantic web, in: *WWW*, 2005.
- [3] K. Anyanwu, A. P. Sheth, P-queries: enabling querying for semantic associations on the semantic web, in: *WWW*, 2003.
- [4] A. Balmin, V. Hristidis, Y. Papakonstantinou, Objectrank: Authority-based keyword search in databases, in: *VLDB*, 2004.
- [5] K. Balog, L. Azzopardi, M. de Rijke, A language modeling framework for expert finding, *Inf. Process. Manage.* 45 (1) (2009) 1–19.
- [6] K. Balog, M. Bron, M. de Rijke, Query modeling for entity search based on terms, categories, and examples, *ACM Trans. Inf. Syst.* 29 (4) (2011) 22.
- [7] H. Bast, A. Chitea, F. M. Suchanek, I. Weber, Ester: efficient search on text, entities, and relations, in: *SIGIR*, 2007.
- [8] V. Bicer, T. Tran, R. Nedkov, Ranking support for keyword search on structured data using relevance models, in: *CIKM*, 2011.
- [9] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, Dbpedia - a crystallization point for the web of data, *J. Web Sem.* 7 (3) (2009) 154–165.
- [10] R. Blanco, P. Mika, S. Vigna, Effective and efficient entity search in rdf data, in: *International Semantic Web Conference* (1), 2011.
- [11] M. Bron, K. Balog, M. de Rijke, Ranking related entities: components and analyses, in: *CIKM*, 2010.
- [12] S. Büttcher, C. L. A. Clarke, B. Lushman, Term proximity scoring for ad-hoc retrieval on very large text collections, in: *SIGIR*, 2006.
- [13] T. H. Cao, T. D. Cao, T. L. Tran, A robust ontology-based method for translating natural language queries to conceptual graphs, in: *ASWC*, 2008.
- [14] P. Castells, M. Fernández, D. Vallet, An adaptation of the vector-space model for ontology-based information retrieval, *IEEE Trans. Knowl. Data Eng.* 19 (2) (2007) 261–272.
- [15] G. Cheng, Y. Qu, Searching linked objects with falcons: Approach, implementation and evaluation, *Int. J. Semantic Web Inf. Syst.* 5 (3) (2009) 49–70.
- [16] T. Cheng, X. Yan, K. C.-C. Chang, Entityrank: Searching entities directly and holistically, in: *VLDB*, 2007.
- [17] J. Chu-Carroll, J. M. Prager, An experimental study of the impact of information extraction accuracy on semantic search performance, in: *CIKM*, 2007.
- [18] J. Chu-Carroll, J. M. Prager, K. Czuba, D. A. Ferrucci, P. A. Duboué, Semantic search via xml fragments: a high-precision approach to ir, in: *SIGIR*, 2006.
- [19] P. Cimiano, P. Haase, J. Heizmann, M. Mantel, R. Studer, Towards portable natural language interfaces to knowledge bases - the case of the orakel system, *Data Knowl. Eng.* 65 (2) (2008) 325–354.
- [20] J. Coffman, A. C. Weaver, A framework for evaluating database keyword search strategies, in: *CIKM*, 2010.
- [21] J. Coffman, A. C. Weaver, Learning to rank results in relational keyword search, in: *CIKM*, 2011.
- [22] C. Comparot, O. Haemmerlé, N. Hernandez, Conceptual graphs and ontologies for information retrieval, in: *ICCS*, 2007.
- [23] O. Corby, R. Dieng-Kuntz, C. Faron-Zucker, Querying the semantic web with corese search engine, in: *ECAI*, 2004.
- [24] D. Damjanovic, M. Agatonovic, H. Cunningham, Natural language interfaces to ontologies: Combining syntactic analysis and ontology-based lookup through the user interaction, in: *ESWC* (1), 2010.
- [25] R. Delbru, N. Toupikov, M. Catasta, G. Tummarello, S. Decker, Hierarchical link analysis for ranking web data, in: *ESWC* (2), 2010.
- [26] L. Ding, R. Pan, T. W. Finin, A. Joshi, Y. Peng, P. Kolari, Finding and ranking knowledge on the semantic web, in: *International Semantic Web Conference*, 2005.
- [27] S. Elbassuoni, M. Ramanath, R. Schenkel, G. Weikum, Searching rdf graphs with sparql and keywords, *IEEE Data Eng. Bull.* 33 (1) (2010) 16–24.
- [28] B. Fazzinga, G. Gianforme, G. Gottlob, T. Lukasiewicz, Semantic web search based on ontological conjunctive queries, *J. Web Sem.* 9 (4) (2011) 453–473.
- [29] M. Fernández, I. Cantador, V. Lopez, D. Vallet, P. Castells, E. Motta, Semantically enhanced information retrieval: An ontology-based approach, *J. Web Sem.* 9 (4) (2011) 434–452.

- [30] S. Ferré, A. Hermann, Semantic search: Reconciling expressive querying and exploratory search, in: *International Semantic Web Conference* (1), 2011.
- [31] H. P. Giger, Concept based retrieval in classical ir systems, in: *SIGIR*, 1988.
- [32] F. Giunchiglia, U. Kharkevich, I. Zaihraye, Concept search, in: *ESWC*, 2009.
- [33] L. Guo, F. Shao, C. Botev, J. Shanmugasundaram, Xrank: Ranked keyword search over xml documents, in: *SIGMOD Conference*, 2003.
- [34] A. Harth, Visinav: A system for visual search and navigation on web data, *J. Web Sem.* 8 (4) (2010) 348–354.
- [35] A. Harth, S. Kinsella, S. Decker, Using naming authority to rank data and ontologies for web search, in: *International Semantic Web Conference*, 2009.
- [36] J. Heflin, J. A. Hendler, S. Luke, Shoe: A blueprint for the semantic web, in: *Spinning the Semantic Web*, 2003.
- [37] P. Heim, T. Ertl, J. Ziegler, Facet graphs: Complex semantic querying made easy, in: *ESWC* (1), 2010.
- [38] T. Hofmann, Probabilistic latent semantic indexing, in: *SIGIR*, 1999.
- [39] A. Hogan, A. Harth, J. Umbrich, S. Kinsella, A. Polleres, S. Decker, Searching and browsing linked data with swse: The semantic web search engine, *J. Web Sem.* 9 (4) (2011) 365–401.
- [40] R. Jelier, M. J. Schuemie, C. C. van der Eijk, M. Weeber, E. M. van Mulligen, B. J. A. Schijvenaars, B. Mons, J. A. Kors, Searching for generifs: Concept-based query expansion and bayes classification, in: *TREC*, 2003.
- [41] X. Jiang, A.-H. Tan, Ontosearch: A full-text search engine for the semantic web, in: *AAAI*, 2006.
- [42] E. Kandogan, R. Krishnamurthy, S. Raghavan, S. Vaithyanathan, H. Zhu, Avatar semantic search: a database approach to information retrieval, in: *SIGMOD Conference*, 2006.
- [43] R. Kaptein, P. Serdyukov, A. P. de Vries, J. Kamps, Entity ranking using wikipedia as a pivot, in: *CIKM*, 2010.
- [44] G. Kasneci, F. M. Suchanek, G. Ifrim, M. Ramanath, G. Weikum, Naga: Searching and ranking knowledge, in: *ICDE*, 2008.
- [45] E. Kaufmann, A. Bernstein, How useful are natural language interfaces to the semantic web for casual end-users?, in: *ISWC/ASWC*, 2007.
- [46] J. Kleb, A. Abecker, Entity reference resolution via spreading activation on rdf-graphs, in: *ESWC* (1), 2010.
- [47] G. Ladwig, T. Tran, Index structures and top-k join algorithms for native keyword search databases, in: *CIKM*, 2011.
- [48] T. M. V. Le, T. H. Cao, S. M. Hoang, J. Cho, Ontology-based proximity search, in: *iiWAS*, 2011.
- [49] G. Li, S. Ji, C. Li, J. Feng, Efficient type-ahead search on relational data: a tastier approach, in: *SIGMOD Conference*, 2009.
- [50] G. Li, B. C. Ooi, J. Feng, J. Wang, L. Zhou, Ease: an effective 3-in-1 keyword search method for unstructured, semi-structured and structured data, in: *SIGMOD Conference*, 2008.
- [51] V. Lopez, V. S. Uren, E. Motta, M. Pasin, Aqualog: An ontology-driven question answering system for organizational semantic intranets, *J. Web Sem.* 5 (2) (2007) 72–105.
- [52] C. Mangold, A survey and classification of semantic search approaches, *IJMSO* 2 (1) (2007) 23–34.
- [53] E. Meij, D. Trieschnigg, M. de Rijke, W. Kraaij, Conceptual language models for domain-specific retrieval, *Inf. Process. Manage.* 46 (4) (2010) 448–469.
- [54] V. M. Ngo, T. H. Cao, Ontology-based query expansion with latently related named entities for semantic text search, in: *Advances in Intelligent Information and Database Systems*, 2010, pp. 41–52.
- [55] Z. Nie, Y. Ma, S. Shi, J.-R. Wen, W.-Y. Ma, Web object retrieval, in: *WWW*, 2007.
- [56] Z. Nie, Y. Zhang, J.-R. Wen, W.-Y. Ma, Object-level ranking: bringing order to web objects, in: *WWW*, 2005.
- [57] J. Pehcevski, A.-M. Vercoustre, J. A. Thom, Exploiting locality of wikipedia links in entity ranking, in: *ECIR*, 2008.
- [58] Y. Qiu, H.-P. Frei, Concept based query expansion, in: *SIGIR*, 1993.
- [59] C. Rocha, D. Schwabe, M. P. de Aragão, A hybrid approach for searching in the semantic web, in: *WWW*, 2004.
- [60] K. Schumacher, M. Sintek, L. Sauer mann, Combining fact and document retrieval with spreading activation for semantic desktop search, in: *ESWC*, 2008.
- [61] J. Stoyanovich, S. J. Bedathur, K. Berberich, G. Weikum, Entity authority: Semantically enriched graph-based authority propagation, in: *WebDB*, 2007.
- [62] F. M. Suchanek, G. Kasneci, G. Weikum, Yago: a core of semantic knowledge, in: *WWW*, 2007.
- [63] T. Tao, C. Zhai, An exploration of proximity measures in information retrieval, in: *SIGIR*, 2007.
- [64] T. Tran, S. Bloehdorn, P. Cimiano, P. Haase, Expressive resource descriptions for ontology-based information retrieval, in: *ICTIR*, 2007.
- [65] T. Tran, D. M. Herzig, G. Ladwig, Semsearchpro - using semantics throughout the search process, *J. Web Sem.* 9 (4) (2011) 349–364.
- [66] T. Tran, T. Mathäß, P. Haase, Usability of keyword-driven schema-agnostic search, in: *ESWC* (2), 2010.
- [67] T. Tran, H. Wang, P. Haase, Hermes: Data web search on a pay-as-you-go integration infrastructure, *J. Web Sem.* 7 (3) (2009) 189–203.
- [68] T. Tran, H. Wang, S. Rudolph, P. Cimiano, Top-k exploration of query candidates for efficient keyword search on graph-shaped (rdf) data, in: *ICDE*, 2009.
- [69] D. Trieschnigg, W. Kraaij, M. J. Schuemie, Concept based document retrieval for genomics literature, in: *TREC*, 2006.
- [70] G. Tummarello, R. Cyganiak, M. Catasta, S. Danielczyk, R. Delbru, S. Decker, Sig.ma: Live views on the web of data, *J. Web Sem.* 8 (4) (2010) 355–364.
- [71] C. J. van Rijsbergen, Towards an information logic, in: *SIGIR*, 1989.
- [72] S. Vassiliadis, G. Triantafyllos, W. Kobrosly, A fuzzy reasoning database question answering system, *IEEE Trans. Knowl. Data Eng.* 6 (6) (1994) 868–882.
- [73] E. M. Voorhees, Using wordnet to disambiguate word senses for text retrieval, in: *SIGIR*, 1993.
- [74] E. M. Voorhees, Query expansion using lexical-semantic relations, in: *SIGIR*, 1994.
- [75] A. Wagner, G. Ladwig, T. Tran, Browsing-oriented semantic faceted search, in: *DEXA* (1), 2011.
- [76] X. Wei, W. B. Croft, Lda-based document models for ad-hoc retrieval, in: *SIGIR*, 2006.
- [77] M. Zhong, X. Huang, Concept-based biomedical text retrieval, in: *SIGIR*, 2006.
- [78] W. Zhou, C. T. Yu, V. I. Torvik, N. R. Smalheiser, A concept-based framework for passage retrieval at genomics, in: *TREC*, 2006.