Information Systems 37 (2012) 294–305

Contents lists available at SciVerse ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/infosys

An ontology-based retrieval system using semantic indexing

Soner Kara, Özgür Alan, Orkunt Sabuncu, Samet Akpınar, Nihan K. Cicekli *, Ferda N. Alpaslan

Department of Computer Engineering, METU, Ankara, Turkey

ARTICLE INFO

Available online 22 September 2011

Keywords:
Semantic web
Keyword-based
Query
Semantic indexing
Ontology

ABSTRACT

In this paper, we present an ontology-based information extraction and retrieval system and its application in the soccer domain. In general, we deal with three issues in semantic search, namely, usability, scalability and retrieval performance. We propose a keyword-based semantic retrieval approach. The performance of the system is improved considerably using domain-specific information extraction, inferencing and rules. Scalability is achieved by adapting a semantic indexing approach and representing the whole world as small independent models. The system is implemented using the state-of-the-art technologies in Semantic Web and its performance is evaluated against traditional systems as well as the query expansion methods. Furthermore, a detailed evaluation is provided to observe the performance gain due to domain-specific information extraction and inferencing. Finally, we show how we use semantic indexing to solve simple structural ambiguities.

1. Introduction

The huge increase in the amount and complexity of reachable information in the World Wide Web caused an excessive demand for tools and techniques that can handle data semantically. The current practice in information retrieval mostly relies on keyword-based search over full-text data, which is modeled as a bag-of-words. However, such a model misses the actual semantic information of the text. In order to deal with this issue, ontologies are proposed [1] for knowledge representation, which are nowadays the backbone of semantic web applications. Both the information extraction and retrieval processes can benefit from such metadata, which gives semantics to plain text.

Once the semantic knowledge is represented via ontologies, the next step is querying the semantic data, also known as semantic search. There are several query languages designed for semantic querying. Currently, SPARQL[1] is the state-of-the-art query language for the Semantic Web. Unfortunately, these formal query languages are not easy to be used by the end-users. Formulating a query using such languages requires the knowledge of the domain ontology as well as the syntax of the language. Therefore, Semantic Web community works on simplifying the process of query formulation for the end-user. The current studies on semantic query interfaces are carried out in four categories; keyword-based, form-based, view-based and natural language-based systems [2]. Among them, keyword-based query interfaces are the most user-friendly ones and people are used to use such interfaces easily, thanks to Google.

Combining the usability of keyword-based interfaces with the power of semantic technologies is one of the most challenging areas in semantic search. According to our vision of Semantic Web, all the efforts towards increasing the retrieval performance while preserving the user-friendliness will eventually come to the point of improving

* Corresponding author. Tel.: +90 312 2105582; fax: +90 312 2105544.
E-mail addresses: soner.kara85@gmail.com (S. Kara), alan@ceng.metu.edu.tr (Ö. Alan), orkunt@ceng.metu.edu.tr (O. Sabuncu), samet@ceng.metu.edu.tr (S. Akpınar), nihan@ceng.metu.edu.tr (N.K. Cicekli), alpaslan@ceng.metu.edu.tr (F.N. Alpaslan).

0306-4379/$ - see front matter © 2011 Elsevier Ltd All rights reserved.
doi:10.1016/j.is.2011.09.004

[1] http://www.w3.org/TR/rdf-sparql-query/.

semantic search with keyword-based interfaces. This is a challenging task as it requires complex queries to be answered with only a few keywords. Furthermore, it should allow the inferred knowledge to be retrieved easily and provide a ranking mechanism to reflect semantics and ontological importance.

In this paper, we present a complete ontology-based framework for the extraction and retrieval of semantic information in limited domains. The system consists of a crawler module, an automated information extraction module, an ontology population module, an inferencing module, and a keyword-based semantic query interface. Our main concern is creating a scalable and user-friendly information retrieval system with high retrieval performance. We applied the framework in the soccer domain and observed the improvements over classical keyword-based approaches. We show that our system achieves very high precision and recall values even for very complex queries in soccer domain. Furthermore, we evaluate and report the effects of different levels of indexing in terms of semantic processing (using only information extraction and using both information extraction and inferencing) on the query performance.

Scalability concerns are divided into two main topics; inferencing and querying. Scalability in terms of inferencing is assured by dividing the whole logical model into individual independent models since inferencing on a single large model is more complex than inferencing on independent smaller models. Similar studies are focused on representing the whole world in a single model. Therefore, they can not fit to large scales.

Scalability in terms of querying is assured by transforming the inferred knowledge into a single special inverted index structure. Using this inverted index structure scales our system up to web search engines, which means answering millions of queries in reasonable time and retrieving information from huge data sources. It also triggers the use of keyword based querying. In this way, the user-friendly way of querying is supported. Studies on ontology based information retrieval use logical queries on ontological models. Thus, their scale is restricted to small sized data sources compared to web scale and logical querying becomes a complex task for ordinary users.

Consequently, our main contribution is a framework which improves the performance of the keyword-based search using semantics without loosing the search scalability. In this aspect, this framework will be a strong base for ontology based search engines with its web scale crawler, information extraction, ontology population and inferencing modules.

The rest of the paper is organized as follows: A brief discussion about the related work is given in Section 2. In Section 3, we give the details of the components of the system, namely information extraction (IE), ontology population, inferencing and information retrieval (IR). In Section 4, we give the evaluation results. Section 5 gives a brief comparison with query expansion methods. Section 6 describes how the system can be extended to support phrasal expressions. In Section 7 we give a brief discussion and Section 8 concludes the paper with some remarks for future work.

2. Related work

The classical or traditional keyword-based information retrieval approaches are based on the vector space model proposed by Salton et al. [3]. In this model, documents and queries are simply represented as a vector of term weights, and the retrieval is done according to the cosine similarity between these vectors. Some of the important studies related to traditional search are [4–7]. This approach does not require any extraction or annotation phases. Therefore, it is easy to implement, however, the precision values are relatively low. The implementation of vector space models in real life applications is provided by the use of tools supporting the inverted index structures such as Lucene. In other words, Lucene like tools connect the real life applications to the theoretical background of vector space models.

The first step towards semantic retrieval is using WordNet synonym sets (synsets) for word semantics [8,9]. The main idea is expanding both indices and queries with the semantics of the words to achieve better recall and precision. If used together with an effective word sense disambiguation (WSD) algorithm, this approach is shown to improve the retrieval performance. On the other hand, a poor WSD will cause degradation in performance. Another drawback of this approach is the lack of complex semantics as it is limited to the relations defined in the WordNet.

Another step towards semantic retrieval is using information extraction. There are many studies on this field. Main dissimilarities between these studies arise from the structure of sources, details of the extracted information and computational/memory resources. NLP-based approaches are domain independent but use parse trees of sentences, pos taggers, chunk parsing, anaphora resolution, etc. in order to extract information. They need heavy computational processes [10–15]. There are some alternative information extraction methods such as pattern/rule-based information extractors against heavy computational costs. These methods are classified according to the creation forms of patterns and rules: automatic or manual. Automatic methods [11,16,17,13,18–21] are superior compared to the manual ones considering the effort spent on the domain. On the other hand, they suffer from low precision-recall rates.

The methods in [22–26] use hand-crafted rules to extract information. Hand-crafted rules are also used in semantic annotation [27–29]. Etzioni et al. [29] uses domain-independent rules to locate individuals of various classes in text. Wessman et al. [28] relies primarily on regular expressions. Cerno is a light-weight framework for semantic annotation of textual documents using domain-specific ontologies [27]. It combines keyword and structure-based annotation rules instead of linguistic patterns. We need a scalable information extractor for a rigid and structured domain. The extractor should also be appropriate for both Turkish and English content. Since the details of the extracted information is crucial for our purposes, we focus on high recall and precision values. Therefore, we have used a hand-crafted method whose details are given in [30].

The studies especially on ontology design, integration and reorganization provide important instruments for ontology based information extraction and retrieval [31–33]. Oberle et al. [31] propose a foundational ontology, SmartSUMO, which is based on DOLCE and SUMO ontologies. The purpose is to integrate the domain ontologies used in SmartWeb. The resulting integrated ontology is described as SmartWeb Integrated Ontology (SWIntO). As SmartSUMO has also linguistic features, SWIntO is used in ontology based textual information extraction. A solid method is proposed regarding ontology design. The textual information is mapped to ontological concepts using a sound method. However, the whole integrated ontology is affected when a new linguistic structure is encountered since linguistic features are part of the foundational ontology. In order to avoid this difficulty, lexical features can be considered separate from the foundational and domain ontologies. Our choice for ontology design does not include a foundational ontology as we do not have an integration purpose for different domains yet. However, we keep the lexical information separate from the ontology by using an information extraction methodology which maps the lexical features to the ontology.

Ontology-based information retrieval systems are discussed in many studies from different aspects [34–36]. The study of [34] which deals with ontology based information retrieval is focused on developing an ontology which makes MPEG-7/21 standards interoperable with domain and application ontologies. An upper ontology is designed for capturing the metadata model of MPEG-7/21 and used in the integration of domain and application ontologies regarding MPEG-7/21. In our study, we developed a single central soccer ontology which enables the representation of the soccer domain knowledge. Our ontology is considered as powerful as the soccer ontology designed in [34] regarding the domain knowledge independent of the interoperability issues. On the other hand, the retrieval methodology proposed in [34] is restricted. As the retrieval methods are based on logical queries, the construction of search queries becomes difficult for naive users. In this context, we propose an index based query answering method enabling the users querying the ontology-based knowledge in an easy and flexible way.

The use of ontologies for information retrieval in organizational memories is discussed in [35]. The importance of meta-level descriptions for structuring, accessing, and maintaining large amounts of heterogeneous information is explored. An ontology-based approach has been proposed for a comprehensive meta-level modeling and retrieval. Ontologies have been classified as domain ontology, information ontology, and enterprise ontology. These are modeled using a conventional object-oriented formalism. According to this classification, we use domain ontology for representing the soccer games. The domain ontology is used to make inferences which are not explicitly extracted from the game summaries. There is no explicit search over ontologies in our work. Queries are answered using inverted indexes constructed out of inferred knowledge, instead.

The interface for querying the ontological knowledge is one of the key points of an information retrieval system. The general approach is storing the extracted data in RDF[2] or OWL[3] format, and querying with RDF query languages such as RDQL or SPARQL. Although this approach offers the ultimate precision and recall performance, it is far from practical since it requires a relatively complex query language. In order to overcome the difficulties of learning a formal query language, a number of query interface methods have been proposed [2]. As we stated earlier, our main focus is keyword-based interfaces. There are several approaches to implement keyword-based querying. To mention a few, SPARK [37] uses a probabilistic query ranking approach for constructing the best query represented by the keywords. Q2Semantic [38] tries to find the best sub-graph expressing the query in the RDF graph. SemSearch [39] uses a template-based approach for query construction. These approaches are not easily scaled to large knowledge bases as they require traversing RDF graphs or querying the same knowledge base several times for a single search.

A scalable alternative to query construction from keywords is semantic indexing. In this approach, semantic data in RDF knowledge bases are indexed in a structured way and made directly available to be used with keyword queries. A similar approach is adapted by [40–42]. They index all the extracted RDF triples together with the corresponding free text. Since they use very basic extraction methods, such a naive indexing seems feasible. However, complex semantics cannot be captured from the indices containing only subject-predicate-object triples as index terms. If a retrieval system should answer complex queries involving extracted and inferred knowledge, the index must be designed and enriched accordingly.

In [43] a dialog based multimodal interface to semantic web in general and its application to the data from Football World Cup 2006 are presented. The emphasis is on forming a question answering system which takes the discourse and context information into account during a dialog based interaction with the user and forms a semantic query to the underlying semantic web components. It is noted that the processing times for some queries may go up to 2 min. Considering the queries related to soccer mentioned in [43], our system not only successfully handles all those queries, but also addresses scalability and performance problems by utilizing a semantic indexing which makes instant query answering possible. Additionally, our ontology individuals are populated automatically by information extraction from web narrations and enriched by applying rule based inferencing.

Our literature survey revealed that current studies on keyword-based semantic searching are not mature enough: either they are not scalable to large knowledge bases or they cannot capture all the semantics in the queries. Our main contribution is to fill this gap by implementing a keyword-based semantic retrieval system using the semantic indexing approach. In other words, we try to implement a system

[2] http://www.w3.org/TR/PR-rdf-syntax/.
[3] http://www.w3.org/TR/owl-guide/.

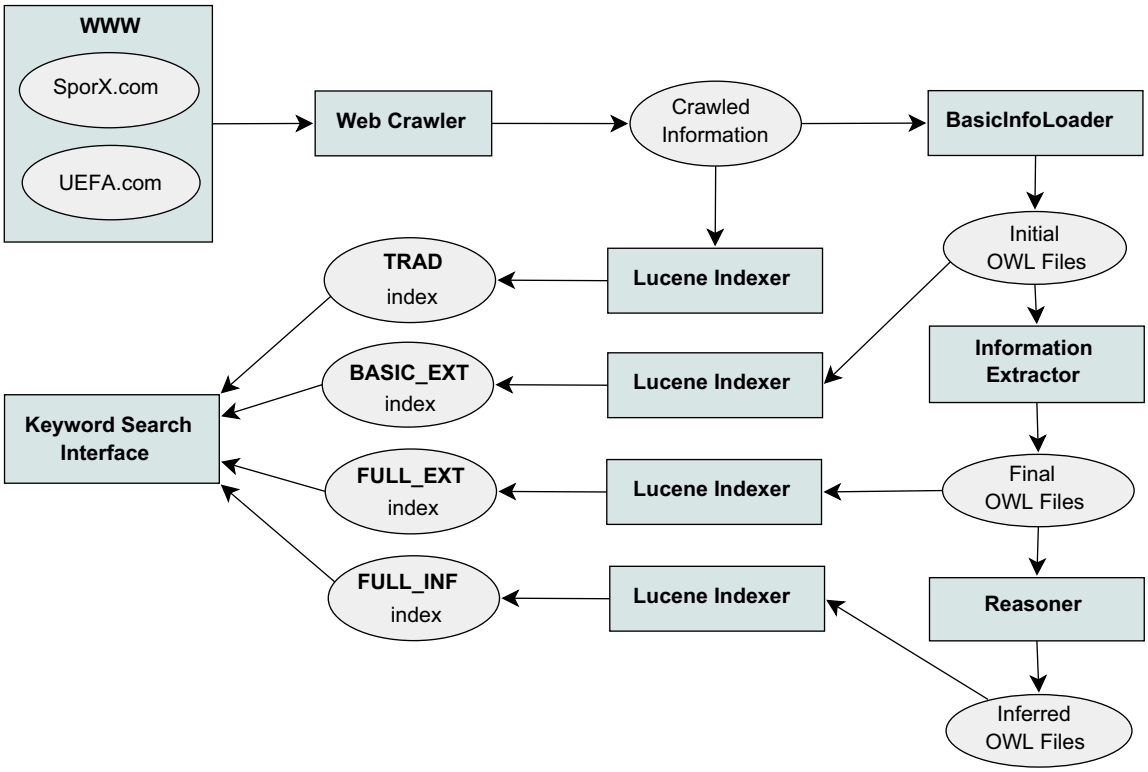

Fig. 1. Overall system diagram.

that performs at least as good as traditional approaches and improves the performance and usability of semantic querying. We tested our system in soccer domain to see the effectiveness of semantic searching over traditional approaches and observed a remarkable increase in precision and recall. Moreover we noted that our system can answer complex semantic queries, which is not possible with the traditional methods. The study presented in this paper can be extended to other domains as well by modifying the current ontology and the information extraction module as described in [30].

3. Our approach to semantic retrieval

Within the scope of this paper we have developed a fully fledged semantic application which (a) contains all aspects of Semantic Web (SW) from information extraction to information retrieval and (b) uses all the cutting-edge technologies such as OWL-DL, inferencing, rules, RDF repositories and semantic indexing. The overall diagram of the framework is shown in Fig. 1. The following sections describe the important aspects of the system starting with a summary of the overall process.

3.1. Overall process

Before giving the details of each module, we find it helpful first to see the overall flow of the system that we adapted for the soccer domain. In the following, we describe the steps we take until the system becomes ready for semantic querying and evaluation.

1. The usable information from web sites such as UEFA[4] and SporX[5] are crawled and temporarily stored. The crawled data contains some basic information such as teams, players, goals, substitutions and the stadium of each soccer game as well as the minute-by-minute narrations of that game in free-text format.
2. Using only these free-text narrations we create an index, `TRAD`, for the traditional keyword search.
3. Using the basic information and narrations we populate the initial OWL files (Section 3.4).
4. From the initial OWL files, we create our second index, `BASIC_EXT`, which contains both the basic information and the narrations. This index is created for evaluation purposes only (i.e, to compare it with the index, `FULL_EXT`, created after the information extraction.)
5. The OWL files created in the previous step are read by the information extractor module. This module populates the OWL files with the extracted events from the narrations such as offsides, fouls, corners, etc. to obtain the final OWL files (Section 3.3).
6. These OWL files are read and indexed to build `FULL_EXT` (Section 3.6.1).

[4] http://www.uefa.com.
[5] http://www.sporx.com.

7. We run the reasoner over these files and obtain new OWL files containing the inferred information (Section 3.5).
8. Finally, we build the index, `FULL_INF`, using these inferred OWLs, which is the final index used in semantic querying.

Although we have focused on the soccer domain in this study, the methodology described above can be adapted to any domain given its ontology. The most domain-specific part of the system is the IE module, which should be extended to deal with other domains. The rest of the paper describes the details of the major components of the system. We start with the design of the domain ontology, then continue with IE and IR components. Finally we report the evaluation results.

3.2. Ontology design

Ontologies are specifications of concepts and relations among them. They play a central role in semantic web applications by providing a shared knowledge about the objects in real world, which promotes reusability and interoperability among different modules. Therefore the quality of the ontology should be the first concern in any semantic application.

For this study, we designed a central soccer ontology, which is utilized by every aspect of the system, especially in the information extraction, inferencing and retrieval phases. We followed an iterative development process in the ontology engineering phase. First, we started with a core ontology including the basic concepts and a simple hierarchy. Then, we experimented with this ontology and fix the issues in reasoning and searching. These steps were repeated until we end up with a stable ontology containing 79 concepts and 95 properties in soccer domain. The full class hierarchy can be seen in Fig. 2.

3.3. Information extraction (IE)

Information extraction is one of the most important parts of ontology-based semantic web applications. It is the process of adding structured information to the knowledge base by processing unstructured resources. In this phase, we use the data crawled from the Web sites such as UEFA and SporX. What we obtain as the output of the web crawler is some basic information specific to a game (teams, players, goals, stadium, etc.) and natural language texts (minute-by-minute narrations of that game). The basic information and the narrations are used as input to our information extraction module. The details of this module are reported in [30]. In this study, we give just an overview of its functionality.

Basically, it is a template-based IE approach for specific domains. Unlike other automated approaches it does not use linguistic tools such as part-of-speech taggers, parsers or phrase chunkers. Therefore our approach can be applied to any domain or any language without using any linguistic tool, although there is the drawback of high effort spent for crafting the templates. The details of the IE module are summarized in two parts: a named entity recognizer and a lexical analyzer.

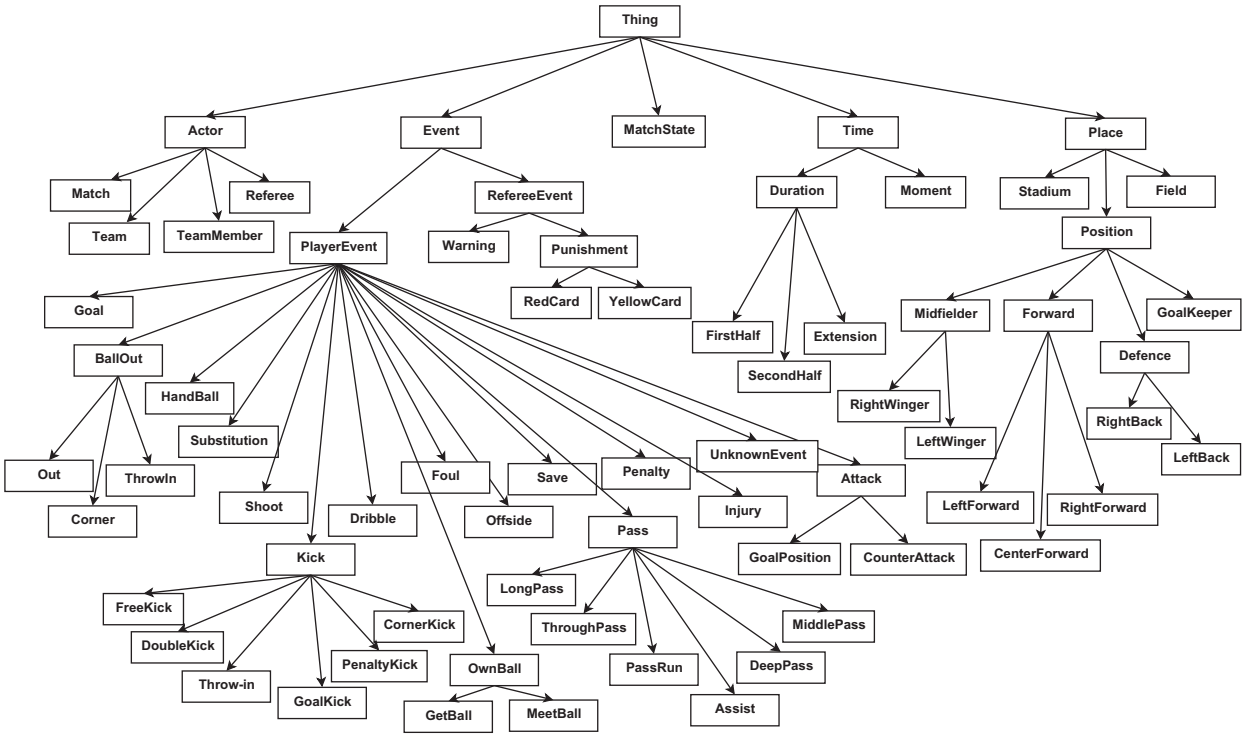

Fig. 2. Domain ontology, class hierarchy.

3.3.1. Named entity recognizer (NER)

As mentioned earlier, IE module takes some basic information such as teams, players, etc., as input in addition to the text narrations. This information is used for recognizing and tagging the named entities in narrations. After running NER, the team and player names are replaced by tags of the form $<team1>$, $<team2>$, $<team1player5>$, etc. For example, the sentence "Iniesta scores!" may become "$<team2player11>$ scores!", if Iniesta is the 11th player of the away team.

3.3.2. Two level lexical analysis

This is the most critical part in our IE module, where complex semantic entities and relations are extracted according to the pre-defined templates. The first level recognizes the defined keywords/phrases and discards the rest. The second level takes the output of the first level as input and extracts information according to the pre-defined templates. According to our survey, most of the studies in Semantic Web lack this kind of extraction as they are usually content with the annotation using only NER.

As reported in [30], we can achieve 100% success rate in UEFA narrations thanks to the language used in the UEFA web-site, which is highly structured and error-free. Fig. 3 gives an idea about the information we can extract from the UEFA web-site. The integration of this module to the system is done in a loosely coupled fashion, so we can use it in semantic applications for any language or domain.

3.4. Ontology population

Ontology population is the process of knowledge acquisition by transforming or mapping unstructured, semi-structured and structured data into ontology individuals [44]. Our information extractor module [30] already does most of the labor by extracting structured information from unstructured text narrations. For example, from the narration "Keita commits a foul after challenging Belletti" we obtain a foul object, more specifically `FOUL(Keita, Belletti)`.

Having the output of the IE module, the ontology population process becomes creating an OWL individual for each object extracted during IE. This is the phase, where the IE module is integrated with the rest of the system. We tried to keep this integration as loose as possible. We deal with this issue at the ontology level by defining some high level properties. Normally, every event in the ontology has its own set of properties. For example, a `Foul` individual has a `punishedPlayer` property, while a goal individual has a `scorerPlayer` property to refer to the subject of the event. The problem is filling the right property of each event by using the information received from the IE module. Our solution to this problem is to define four generic properties for each event in our ontology, namely `subjectPlayer`, `objectPlayer`, `subjectTeam` and `objectTeam`. These properties are generic and have sub-properties special to each event. For example `Goal` event has the property `scorerPlayer`, which is in turn a sub-property of `subjectPlayer` property. In this way, we can automatically fill in the `scorerPlayer` property of a `Goal` event by using the subject of the event. Similarly `injuredPlayer` property of an `Injury` event will be filled in with the object of the event, since `injuredPlayer` property is defined as a sub-property of `objectPlayer` property in the ontology.

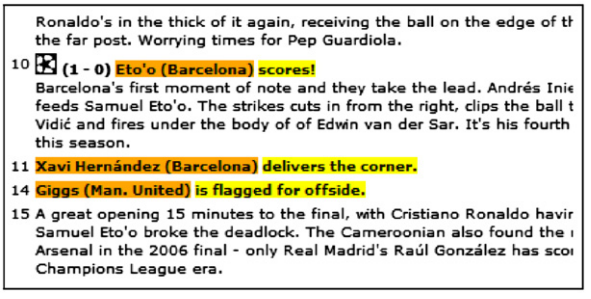
Ronaldo's in the thick of it again, receiving the ball on the edge of th
the far post. Worrying times for Pep Guardiola.
10 (1 - 0) Eto'o (Barcelona) scores!
Barcelona's first moment of note and they take the lead. Andrés Inie
feeds Samuel Eto'o. The strikes cuts in from the right, clips the ball t
Vidić and fires under the body of of Edwin van der Sar. It's his fourth
this season.
11 Xavi Hernández (Barcelona) delivers the corner.
14 Giggs (Man. United) is flagged for offside.
15 A great opening 15 minutes to the final, with Cristiano Ronaldo havir
Samuel Eto'o broke the deadlock. The Cameroonian also found the
Arsenal in the 2006 final - only Real Madrid's Raúl González has sco
Champions League era.

Fig. 3. Example extractions from UEFA narrations.

If the IE module cannot extract any property of an event, we still create an individual with empty properties. Thus, the recall performance for simple queries will not be affected even if the IE fails to extract some details of the event. Moreover, if no event is detected in a narration, an individual with the type `UnknownEvent` is created for that narration. Unknown events are not discarded because of the reasons described in Section 3.6.1. Fig. 4 shows the process of ontology population starting from UEFA narrations ending with OWL individuals.

Ontology population is not restricted with the events extracted from the IE module. As mentioned earlier, the crawled data also contains some basic information about the game including players, teams, referees, stadium, etc. This information is also added to the ontology by creating an OWL individual for each of them if they do not already exist in the knowledge base.

3.5. Inferencing and rules

The formal specification of Web Ontology Language, OWL, is highly influenced by Description Logics (DLs). OWL-DL is designed to be computationally complete and decidable version of OWL, thus it benefits from a wide range of sound, complete and terminating DL reasoners. For our inferencing module, we use Pellet,[6] an open-source DL-reasoner, which supports all the standard inferencing services such as consistency checking, concept satisfiability, classification and realization.

Consistency checking ensures that there is no contradictory assertion in the ontology. In order to benefit from this feature, we specify some property restrictions during the ontology development. There are two kinds of restrictions in OWL: value constraints and cardinality constraints. We use value constraints, for example, to state that only the goalkeepers (a subset of players) are allowed in the position of goalkeeping and using a cardinality constraint, we can say that only one goalkeeper is allowed in the game. These restrictions not only are useful in consistency checking but also allow new information to be inferred. For example, we

[6] http://clarkparsia.com/pellet (last visited on 06/07/2010).

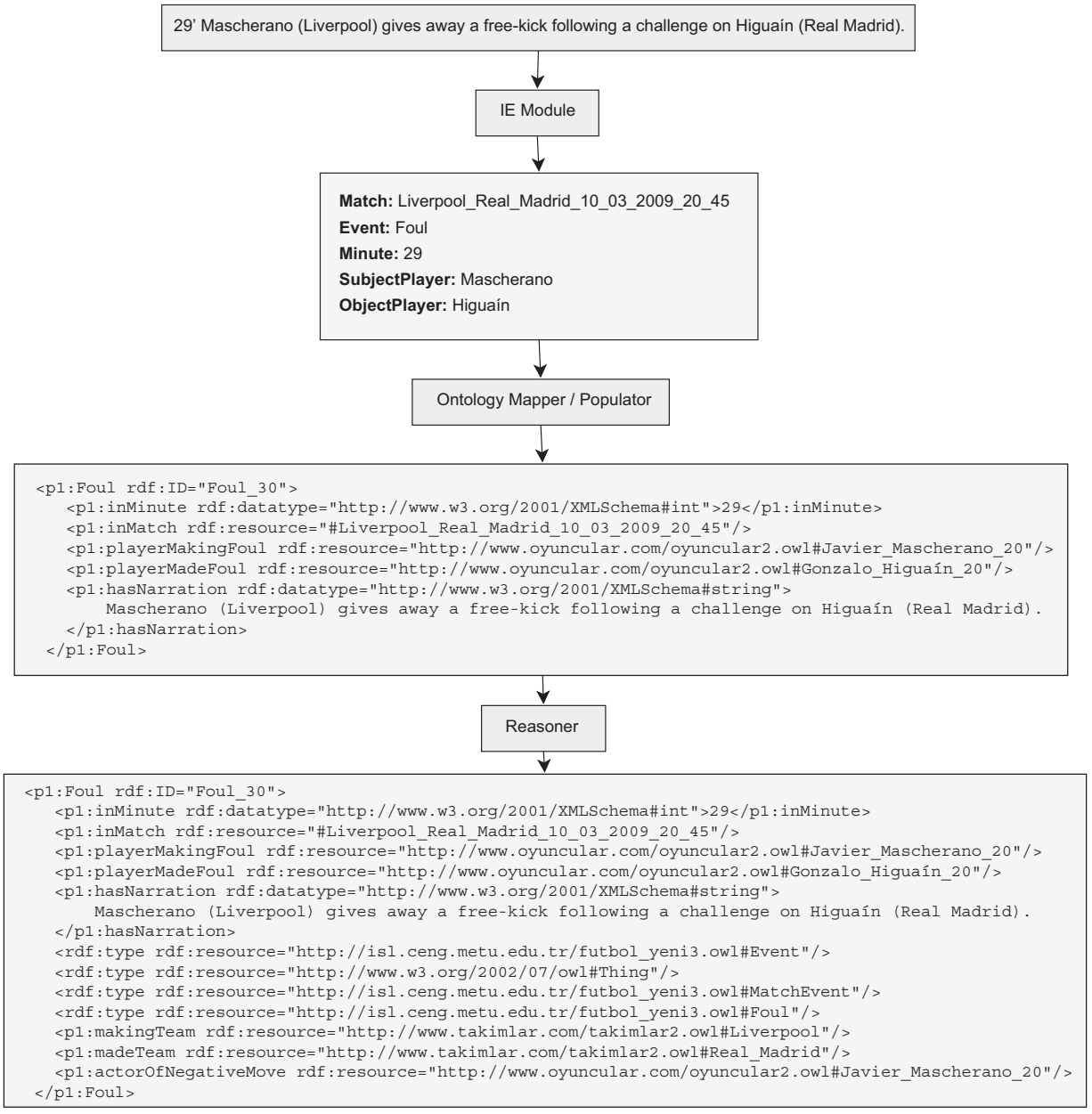

Fig. 4. Information extraction and ontology population.

could infer the type of an individual if it is the value of a property whose range is restricted to a certain class.

Using classification reasoning we obtain the whole class hierarchy according to class-subclass definitions in the ontology. Inferring new knowledge through classification is a domain independent process and its contribution to the knowledge base is trivial. A simple example can be seen in Fig. 5, where the class hierarchy of "Long Pass" is inferred.

In order to infer more interesting information, we use Jena[7] rules. To illustrate the power of Jena rules, we give the example of inferring an "Assist" event. Using the Jena rule shown in Fig. 6, we are able to add a new "Assist" individual to our knowledge base. The rule simply looks for two events, namely a "Goal" and a "Pass" that happened in the same soccer game in the same minute and the receiver of the pass is the same person with the scorer. If this is the case, then an "Assist" individual is created and added to the knowledge base.

Inferencing is a costly process, especially if the number of assertions (ABox) is large. This problem should be handled carefully to maintain the scalability of the system. Therefore we take some measures in order to deal with this issue. First of all, we keep each soccer game separate

[7] http://jena.sourceforge.net/ (last visited on 06/07/2010).

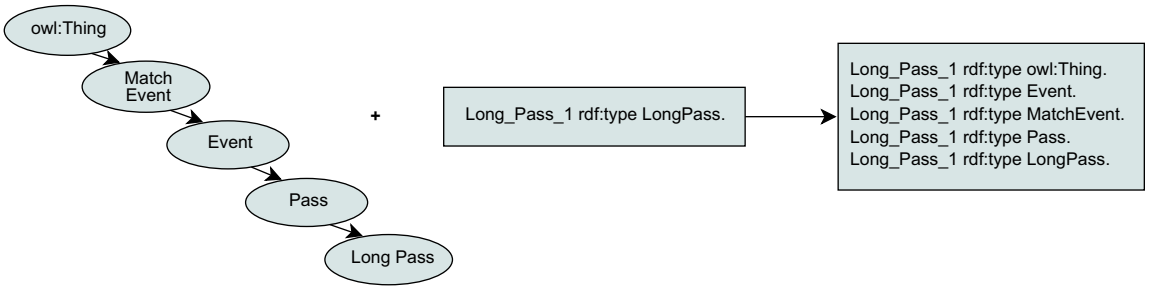

Fig. 5. Inferring class hierarchy of long pass.

```
noValue (?pass rdf:typepre:Assist)
(?pass rdf:type pre:Pass)
(?pass pre:passingPlayer ? passer)
(?pass pre:passReceiver ?receiver)
(?pass pre:inMatch ?match)
(?pass pre:inMinute ?minute)
(?goal pre:inMatch ?match)
(?goal pre:inMinute ?minute)
(?goal  pre:scorerPlayer ?receiver)
makeTemp (?tmp)

->  (?tmp rdf:type pre:Assist)
   (?tmp pre:inMatch ?match)
   (?tmp pre:inMinute ?minute)
   (?tmp pre:passingPlayer ?passer)
   (?tmp pre:passReceiver ?receiver)
```

Fig. 6. An example for Jena rules (assist rule).

Table 1
Index structure (simplified for better understanding).

Field	Value
docNo	7
Event	Foul
Match	Chelsea_Barcelona_06_05_2009_20_45
Team1	Chelsea
Team2	Barcelona
Date	2009-05-06
Minute	43
subjectPlayer	Michael Ballack
subjectTeam	–
objectPlayer	Sergio Busquets
objectTeam	–
Narration	Ballack gives away a free-kick following a challenge on Busquets

from each other and run the inferencing separately. Then, we disjunctively add the inferred information to the knowledge base. So, the time needed for the inferencing of a soccer game becomes independent of the total number of games. Second, all reasoning tasks, including classification and inferencing, are done offline, i.e. prior to querying. This improves scalability as well as the query efficiency since no online reasoning is needed at runtime.

3.6. Semantic indexing and retrieval

There are many methods to query a semantic knowledge base and retrieve the results. These methods are reviewed by [2] in four categories, namely keyword-based, natural language-based, view-based and form-based semantic querying. Out of these, keyword-based interfaces provide the most comfortable and relaxed way of querying for the end-user. Other methods, although they allow more precise queries to be formulated, require more user interaction depending on the size of the domain. Although keyword-based interfaces have their own disadvantages such as ambiguities, there are ways to minimize them as we will mention in Section 6. Now, having decided on keyword-based querying, the next question is how we achieve high retrieval performance and scalability. The answer is, simply, semantic indexing.

As we mentioned in our literature survey, current keyword-based approaches either do real-time traversals in large RDF graphs or make simple RDF triple indexing. In other words, they do not focus on both scalability and retrieval performance. We propose a model, called semantic indexing, that extends traditional keyword-search with extracted and inferred information using domain ontology. The indexing mechanism is built upon Apache Lucene,[8] a scalable and high performance indexer and searcher, which is essentially designed for free-text search. Semantic retrieval is achieved by implementing a custom ranking for Lucene indices so that documents containing ontological information get higher rates. The details of the index structure and ranking are given in Section 3.6.1 and 3.6.2 respectively.

3.6.1. Index structure

The structure of the semantic index has utmost importance in the retrieval performance. We constructed a Lucene index such that each entry represents a soccer event. As we have mentioned in the previous sections, each event has its own properties associated with it, such as subjects and objects. That information is also included with each event. We also include full-text narrations associated with events to the index. This is especially important if the event type is unknown (an event which is not recognized by the information extractor). Adding full-text narrations to the index tolerates the incomplete event information, thus ensures at least the recall values of the traditional full-text search. The index structure can be seen with an example entry in Table 1.

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

For the inferred OWL files, we build an extended version of this index. In addition to the basic information contained in this index, the inferred index also contains a field for all the inferred types of an event, a field for inferred player properties and a field to keep inferred information according to semantic rules. The additional information appended to the inferred index can be seen in Table 2. Note that the `subjectTeam` and `objectTeam` fields are also filled using the semantic rules.

3.6.2. Searching and ranking

In the traditional keyword search, the indexed documents usually contain nothing but raw text associated with that document. Lucene can easily handle such indices and its default ranking gives usually good results. However, complex indices should be handled carefully. In order to take the advantages of our ontology-aided index structure, we slightly modified the default querying and ranking mechanism of Lucene. First of all, we boosted the ranking of fields containing the extracted and inferred information to stress the importance of them. Second, these fields are re-ranked according to their importance. For example, the "event" field is given the highest ranking. This approach prevents misleading stemming from ambiguous words in full-text. For example, suppose a narration contains "Ronaldo misses a goal". Searching for a "goal" in a traditional search may return this document in the first place, which is a false positive. However, in the ontology-aided index, the events whose type is `Goal` will have higher ranks. Since the type of the event above is a `Miss`, it will have a lower rank.

4. Evaluation

In order to evaluate the retrieval performance of our system, we have crawled 10 UEFA matches, containing a total of 1182 narrations. Out of these narrations, our IE module was able to extract 902 events. Using these data, we constructed four Lucene indices for detailed comparisons. First, we built a traditional full-text index, `TRAD`, using only the narrations of the UEFA matches. This index is used as the baseline for the performance of other methods. Then, we built two indices for the ontology aided semantic search, namely `BASIC_EXT` and `FULL_EXT`, where the former contains only the basic information available in the UEFA crawl and the latter contains the extracted information in addition to the basic information. Finally, we built an index, `FULL_INF`, which is the expanded version of `FULL_EXT` with the inferred knowledge. All of the indices are evaluated with the queries shown in Table 3.

Table 2
Additional information in inferred index.

Field	Value
Event	Negative event foul
`subjectPlayerProp`	Left back defence player
`subjectTeam`	Chelsea
`objectPlayerProp`	Center forward player
`objectTeam`	Barcelona
FromRules	–

Table 3
Evaluation queries.

Q-1	Find all goals (**query**: **goal**)
Q-2	Find all goals scored by Barcelona (**query**: **barcelona goal**)
Q-3	Find all goals scored by Messi at Barcelona (**query**: **messi barcelona goal**)
Q-4	Find all punishments (**query**: **punishment**)
Q-5	Find all yellow cards received by Alex (**query**: **alex yellow card**)
Q-6	Find all goals scored to Casillas (**query**: **goal scored to casillas**)
Q-7	Find all negative moves of Henry (**query**: **henry negative moves**)
Q-8	Find all events involving Ronaldo (**query**: **ronaldo**)
Q-9	Find all saves done by the goalkeeper of Barcelona (**query**: **save goalkeeper barcelona**)
Q-10	Find all shoots delivered by defence players (**query**: **shoot defence players**)

The results can be seen in Table 4. First of all, consider the first three queries. There is a considerable difference between `TRAD` and the other methods. The reason is that UEFA narrations use the phrase "P scores!" when the player P scores a goal. Since they omit the word "goal" in narrations, the traditional index is not able to retrieve all the goals with the keyword query: "goal". However, the information extraction module can successfully recognize the goal and we can index it as a document with its `eventType` field filled as "goal". Thus, the improved index can answer both the queries "goal" and "scores" successfully. That is the reason why `BASIC_EXT` and the other indices have very high precision rates.

The improvement provided by the information extraction module can be seen clearly by looking at the difference between `BASIC_EXT` and `FULL_EXT` in 9th and 10th queries. The difference stems from the extracted events such as shoots and goalkeeper saves.

The improvements stemming from the inferencing can be observed by looking at the queries 4, 7 and 10. In these queries, `FULL_INF` index performs much better than other indices, because it contains additional information due to ontological inferencing and classification. For example, the 4th query exploits the inferred knowledge about the fact that red cards and yellow cards are also known as punishments. Similarly, the 10th query benefits from the inferred defence players through classification. Finally, the 7th query uses the knowledge obtained from the property hierarchies defined in the ontology. This means, the system can recognize the properties such as `actorOfMissedGoal`, `actorOfOffside`, and `actorOfRedCard` as `actorOfNegativeMove`. Moreover, in the 6th query, we can see the effect of Jena rules. Here, according to one of the rules we defined, we can infer the implicit knowledge of which goal is scored to which goalkeeper, even if that knowledge does not exist explicitly.

If we look at the query 8, we can see that all the four indices perform nearly the same. The reason is that, it is a simple query with a single player name (Ronaldo). So, it does not contain much information that the semantic indexing can make use of. However, even in such queries,

Table 4
Evaluation results (mean average precision).

Queries	TRAD		BASIC_EXT		FULL_EXT		FULL_INF	
Q-1	0.5/35	1.4%	35/35	100%	35/35	100%	35/35	100%
Q-2	0.4/7	5.7%	5.3/7	75.7%	5.3/35	75.7%	5.3/35	75.7%
Q-3	0.7/3	23.3%	3/3	100%	3/3	100%	3/3	100%
Q-4	0/43	0%	0/43	0%	0/43	0%	43/43	100%
Q-5	1.1/2	55%	2/2	100%	2/2	100%	2/2	100%
Q-6	0.1/9	1.1%	5.7/9	63.3%	5.6/9	62.2%	9/9	100%
Q-7	2.2/7	31.4%	1.9/7	27.1%	2.3/7	32.8%	6.3/7	90.0%
Q-8	7.9/11	71.8%	8.6/11	78.1%	8.5/11	77.2%	7.4/11	75.9%
Q-9	5.1/8	63.7%	4.5/8	56.2%	6.3/8	78.7%	7.5/8	93.7%
Q-10	0/83	0%	0/83	0%	21.9/83	26.4%	81.4/83	98.1%

the performance does not drop below the traditional approach, because the full-text narrations are not discarded but preserved in a separate field. In other words, our approach guarantees at least the performance of traditional approach in the worst case.

Our evaluation results show that starting from the BASIC_EXT, each index makes a solid improvement over its predecessor and we ultimately reach the desired performance in FULL_INF. Note that the performances of BASIC_EXT and FULL_EXT are also satisfactory. Especially the huge gap between the TRAD and BASIC_EXT is important, because it shows that even the basic information provided in the crawled data can make a great difference. However, when the queries get more and more complex, we need domain-specific information extraction, rules and inferencing to handle them. So, with this framework, we provide a set of opportunities for the developer to tweak their system according to the user needs. In any case, the framework guarantees to provide the same scalability and user-friendliness.

5. Comparison with query expansion

In Section 4 we have compared our solution with traditional approaches and observed a remarkable improvement. However, one can still ask whether this result could be achieved by using only the query expansion methods. Therefore, a final experiment showing the difference between our solution and the query expansion methods is needed. To fill this gap we implemented a prototype for the query expansion which uses the domain terms to extend queries. For example, a query containing the word "goal" is expanded with the verbs "score", "miss" and their derivatives. Ontological information is also used for query expansion, thus the query "punishment" is augmented with its subclasses such as "yellow card" and "red card" as well as the verb "book" and its derivatives. The expanded queries are run directly on free-text, so we can clearly see the improvements stemming from the query expansion and compare it with our solution.

The results of the experiment are shown in Table 5. At a glance, we can easily say that the performance of the query expansion method resides between the traditional approach and the semantic indexing as we expected. The effects of the query expansion can be seen clearly by looking at the queries 1, 2, 3 and 4. The first three queries are improved by the expansion with the term "scores". The huge increase in the 4th query stems from the ontological expansion of the query term "punishment". Still, the performance is not close to our solution with semantic indexing. The rest of the queries are not much affected by the query expansion method due to the absence of suitable expansion terms. Some queries are even deteriorated by this method because of the false positives introduced by the extra query terms.

Table 5
Comparison with query expansion.

Queries	TRAD (%)	QUERY_EXP (%)	FULL_INF (%)
Q-1	1.4	30.1	100
Q-2	5.7	16.4	75.7
Q-3	23.3	49.0	100
Q-4	0	63.6	100
Q-5	55	51.5	100
Q-6	1.1	11.5	100
Q-7	31.4	27.16	90.0
Q-8	71.8	71.8	75.9
Q-9	63.7	62.5	93.7
Q-10	0	4.3	98.1

So, we conclude that although the query expansion method can improve the performance of traditional approaches, it cannot exceed the performance of semantic indexing, because it is unable to capture the actual semantics of the query words.

6. Adding phrasal expression support

We have mentioned in Section 3.6 that the ambiguity problem is the most important issue in keyword-based search interfaces. Ambiguities arise usually in two forms: lexical and structural. Lexical ambiguities are caused by homonyms and structural ambiguities occur when a sentence or phrase imply more than one meaning due to the multiple assignments of the same word. We can solve the lexical ambiguities only if the information extraction module recognizes them since we did not implement a word disambiguation method. Structural ambiguities, however, can be solved in the semantic indexing phase. For demonstration purposes, we solve simple structural ambiguities by adding phrasal expression support to our current implementation.

Table 6
Effects of phrasal expressions.

Query	FULL_INF (%)	PHR_EXP (%)
Foul by Daniel	48.2	100
Foul by Daniel to florent	47.7	100
Foul by florent to Daniel	100	100

Suppose a user tries to find the fouls that Alex made to Ronaldo and types the query "foul Alex Ronaldo". The system may also retrieve the fouls by Ronaldo to Alex because both of the players can act either as a subject or an object. To solve this structural ambiguity, we introduce simple phrasal expressions such as "to X", "by X", "of X", so that the system can understand the intended object and subject of the query. The cost of the implementation was negligible as it is achieved just by adding two new fields, one for the subject and one for the object. The content of each field is the concatenation of the name of the object or subject with the corresponding preposition.

We compared the performance of the new index (`PHR_EXP`) with the original one (`FULL_INF`). We prepared three artificial queries for this purpose. The first query measures the subject discrimination performance (Daniel is the maker of foul). The second query adds an object (Florent) to the first query and finally the object and subject positions are swapped in the third query. The results are shown in Table 6. Note that the old index has difficulties with handling the ambiguities. It cannot discriminate which player is the object and which player is the subject of the query, but constantly assumes Florent as the subject player without any reason. The new index, on the other hand, can successfully discriminate the object and the subject in every condition.

7. Discussion

This paper shows how we achieve semantic querying without compromising from scalability and user-friendliness. There is another important benefit of our approach that we did not cover in this study, which is flexibility, i.e. how easily the system responds to data updates or modifications. We claim that the use of the semantic index as an upper layer above ontology makes the knowledge base much more flexible.

Today, most of the semantic applications use ontologies as the main data structure of the system, i.e. all the heavy read/write operations are run directly over the ontology individuals. However, ontologies and ontology individuals are not meant to be used in such conditions. By definition, they are strict, formal representations of knowledge and assertions. In other words, they are immutable by nature and are not optimized for rapid changes or updates. Therefore we recommend an efficient secondary data storage for these tasks, such as semantic indexing as our solution does.

To illustrate how the semantic indexing improves the flexibility, we can give the following example. Suppose we want to add support for another language in the query interface, so that we can query the same knowledge base with two different languages. To achieve this with ontology individuals, one has to either duplicate the individuals for the second language or duplicate the properties with the translated values. Either solution is impractical when the number of individuals is too large. With the semantic indexing, however, it is as easy as adding the translated value next to its original value for each field. Expanding the index terms with WordNet synonyms, natural language phrases or even word stems are other examples that can be achieved easily with semantic indexing.

8. Conclusion and future work

We have presented a novel semantic retrieval framework and its application in the soccer domain, which includes all the aspects of Semantic Web, namely, ontology development, information extraction, ontology population, inferencing, semantic rules, semantic indexing and retrieval. When these technologies are combined with the comfort of keyword-based search interface, we obtain a user-friendly, high performance and scalable semantic retrieval system. The evaluation results show that our approach can easily outperform both the traditional approach and the query expansion methods. Moreover, we observed that the system can answer complex semantic queries without requiring formal queries such as SPARQL. We observe that the system can get close to the performance of SPARQL, which is the best that can be achieved with semantic querying. Finally, we show how the structural ambiguities can be resolved easily using semantic indexing.

The current implementation can be extended and improved in many ways. First of all, we are planning to enrich the knowledge base to support multiple languages as we mentioned in Section 7. The performance will be further improved by implementing a word disambiguation module for lexical ambiguities. Finally, a mechanism that expands the index automatically according to the user feedback is one of our future goals.

Acknowledgements

This work is partially supported by The Scientific and Technical Council of Turkey Grant TUBITAK EEEAG-107E234 and by TUBITAK TEYDEB-3080231.

References

[1] T.R. Gruber, Toward principles for the design of ontologies used for knowledge sharing, International Journal of Human Computer Studies 43 (1995) 907–928.
[2] V. Uren, Y. Lei, V. Lopez, H. Liu, E. Motta, M. Giordanino, The usability of semantic search tools: a review, Knowledge Engineering Review 22 (2007) 361–377.
[3] G. Salton, A. Wong, C.S. Yang, A vector space model for automatic indexing, Communications of the ACM 18 (1975) 613–620.
[4] K.S. Jones, A statistical interpretation of term specificity and its application in retrieval, Journal of Documentation 28 (1972) 11–21.
[5] G. Salton, M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, 1983.
[6] G. Salton, E.A. Fox, H. Wu, Extended boolean information retrieval, Communications of the ACM 26 (1983) 1022–1036.

[7] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Information Processing and Management: An International Journal 24 (1988) 513–523.
[8] J. Gonzalo, F. Verdejo, I. Chugur, J. Cigarrin, Indexing with wordnet synsets can improve text retrieval, Computing Research Repository (1998) 38–44.
[9] R. Mihalcea, D. Moldovan, Semantic indexing using wordnet senses, in: Proceedings of the ACL-2000 Workshop on Recent advances in Natural Language Processing and Information Retrieval, Association for Computational Linguistics, Morristown, NJ, USA, 2000, pp. 35–45.
[10] M. Banko, O. Etzioni, The tradeoffs between open and traditional relation extraction, in: Proceedings of ACL-08: HLT, Association for Computational Linguistics, Columbus, Ohio, 2008, pp. 28–36.
[11] R.M. Casado, E. Alfonseca, P. Castells, Automatic extraction of semantic relationships for WordNet by means of pattern learning from Wikipedia, in: Tenth International Conference on Applications of Natural Language to Information Systems, NLDB 2005, Springer, 2005, pp. 67–79.
[12] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D.S. Weld, A. Yates, Web-scale information extraction in knowitall: (preliminary results), in: WWW '04: Proceedings of the 13th International Conference on World Wide Web, ACM, New York, NY, USA, 2004, pp. 100–110.
[13] M. Surdeanu, M. Ciaramita, Robust information extraction with perceptrons, in: Proceedings of the NIST 2007 Automatic Content Extraction Workshop ACE07.
[14] A. Kiryakov, B. Popov, I. Terziev, D. Manov, D. Ognyanoffe, Semantic annotation, indexing, and retrieval, Journal of Web Semantics (2005).
[15] D. Ferruci, A. Lally, Uima: an architectural approach to unstructured information processing in the corporate research environment, Natural Language Engineering 10 (2004) 327–348.
[16] R.J. Mooney, R. Bunescu, Mining knowledge from text using information extraction, SIGKDD Explorations Newsletter 7 (2005) 3–10.
[17] A. Schutz, P. Buitelaar, Relext: a tool for relation extraction from text in ontology extension, in: The Semantic Web (ISWC 2005), 2005, pp. 593–606.
[18] F. Ciravegna, S. Chapman, A. Dingli, Y. Wilks, Learning to harvest information for the semantic web, in: European Semantic Web Symposium/Conference, vol. 3053, 2004, pp. 312–326.
[19] I. Muslea, S. Minton, C. A. Knoblock, Active learning with strong and weak views: a case study on wrapper induction, in: International Joint Conference on Artificial Intelligence, 2003, pp. 415–420.
[20] D. Freitag, N. Kushmerick, Boosted wrapper induction, in: Seventeenth National Conference on Artificial Intelligence, 2000, pp. 577–583.
[21] P. Cimiano, S. Handschuh, S. Staab, Towards the self-annotating web, in: Thirteenth World Wide Web Conference Series, 2004, pp. 462–471.
[22] N. Nitta, N. Babaguchi, T. Kitahashi, Extracting actors, Proceedings Fifteenth International Conference on Pattern Recognition, 2000, vol. 4, 2000, pp. 718–721.
[23] C. Ramakrishnan, K. Kochut, A.P. Sheth, A framework for schema-driven relationship discovery from unstructured text, in: International Semantic Web Conference, 2006, pp. 583–596.
[24] H. Saggion, H. Cunningham, K. Bontcheva, D. Maynard, O. Hamza, Y. Wilks, Multimedia indexing through multi-source and multi-language information extraction: The MUMIS project, Data and Knowledge Engineering 48 (2004) 247–264.
[25] C. Xu, J. Wang, K. Wan, Y. Li, L. Duan, Live sports event detection based on broadcast video and web-casting text, in: MULTIMEDIA '06: Proceedings of the Fourteenth annual ACM international conference on Multimedia, ACM, New York, NY, USA, 2006, pp. 221–230.
[26] Y. Yang, L. Li, Research on sports game news information extraction, in: International Conference on Natural Language Processing and Knowledge Engineering, NLP-KE 2007, 2007, pp. 96–101.
[27] N. Kiyavitskaya, N. Zeni, J.R. Cordy, L. Mich, J. Mylopoulos, Cerno: light-weight tool support for semantic annotation of textual documents, Data and Knowledge Engineering 68 (2009) 1470–1492.
[28] A. Wessman, S. W. Liddle, D. W. Embley, A generalized framework for an ontology-based data-extraction system, in: Fourth International Conference on Information Systems Technology and its Applications, 2005, pp. 239–253.
[29] O. Etzioni, M.J. Cafarella, D. Downey, A. maria Popescu, T. Shaked, S. Soderland, D.S. Weld, A. Yates, Unsupervised named-entity extraction from the web: an experimental study, Artificial Intelligence 165 (2005) 91–134.
[30] D. Tunaoglu, O. Alan, O. Sabuncu, S. Akpinar, N.K. Cicekli, F.N. Alpaslan, Event extraction from Turkish football web-casting texts using hand-crafted templates, in: Proceedings of the 2009 IEEE International Conference on Semantic Computing, ICSC '09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 466–472.
[31] D. Oberle, A. Ankolekar, P. Hitzler, P. Cimiano, M. Sintek, M. Kiesel, B. Mougouie, S. Baumann, S. Vembu, M. Romanelli, P. Buitelaar, R. Engel, D. Sonntag, N. Reithinger, B. Loos, H.-P. Zorn, V. Micelli, R. Porzel, C. Schmidt, M. Weiten, F. Burkhardt, J. Zhou, DOLCE ergo SUMO: on foundational and domain models in the SmartWeb integrated ontology (SWIntO), Journal of Web Semantics 5 (2007) 156–174.
[32] A. Gangemi, C. Catenacci, M. Battaglia, Inflammation ontology design pattern: an exercise in building a core biomedical ontology with descriptions and situations, in: D.M. Pisanelli (Ed.), Ontologies in Medicine, IOS Press, Amsterdam, 2004.
[33] J. Hunter, Adding multimedia to the semantic web—building an MPEG-7 ontology, in: Proceedings of the First Semantic Web Working Symposium, 2001, pp. 261–281.
[34] C. Tsinaraki, P. Polydoros, S. Christodoulakis, Interoperability support between mpeg-7/21 and owl in ds-mirf, IEEE Transactions on Knowledge and Data Engineering 19 (2007) 219–232.
[35] M. Liao, A. Abecker, A. Bernardi, K. Hinkelmann, M. Sintek, Ontologies for knowledge retrieval in organizational memories, in: Proceedings of the Workshop on Learning Software Organizations, Fraunhofer Institute for Experimental Software Engineering, 1999, pp. 11–25.
[36] H.-M. Muller, E.E. Kenny, P.W. Sternberg, Textpresso: an ontology-based information retrieval and extraction system for biological literature, PLoS Biology 2 (2004) e309.
[37] Q. Zhou, C. Wang, M. Xiong, H. Wang, Y. Yu, Spark: adapting keyword query to semantic search, Proceedings of the 6th International Semantic Web Conference and Second Asian Semantic Web Conference (ISWC/ASWC2007), Busan, South Korea, Lecture Notes in Computer Science, vol. 4825, Springer Verlag, Berlin, Heidelberg, 2007, pp. 687–700.
[38] H. Wang, K. Zhang, Q. Liu, T. Tran, Y. Yu, Q2semantic: a lightweight keyword interface to semantic search, in: ESWC, Springer, 2008, pp. 584–598.
[39] Y. Lei, V.S. Uren, E. Motta, Semsearch: a search engine for the semantic web, 2006, pp. 238–245.
[40] U. Shah, T. Finin, A. Joshi, R.S. Cost, J. Matfield, Information retrieval on the semantic web, in: CIKM '02: Proceedings of the Eleventh International Conference on Information and Knowledge Management, ACM, New York, NY, USA, 2002, pp. 461–468.
[41] J. Davies, R. Weeks, Quizrdf: search technology for the semantic web, Hawaii International Conference on System Sciences 4 (2004) 40112–40120.
[42] I. Celino, E.D. Valle, D. Cerizza, A. Turati, Squiggle: an experience in model-driven development of real-world semantic search engines, in: L. Baresi, P. Fraternali, G.-J. Houben (Eds.), ICWE, Lecture Notes in Computer Science, vol. 4607, Springer, 2007, pp. 485–490.
[43] D. Sonntag, R. Engel, G. Herzog, A. Pfalzgraf, N. Pfleger, M. Romanelli, N. Reithinger, Smart web handheld – multimodal interaction with ontological knowledge bases and semantic web services, in: Proceedings of the International Workshop on AI for Human Computing (In Conjunction with IJCAI), 2007.
[44] Semantic Web, The Semantic Web Wiki, 2008.