Sport News Semantic Search with Natural Language Questions

Quang-Minh Nguyen[1(✉)], Son-Hong Ngo[2], and Tuan-Dung Cao[2]

[1] School of Electronics and Telecommunications,
Hanoi University of Science and Technology, Hanoi, Vietnam
minh.nguyenquang@hust.edu.vn
[2] School of Information and Communication Technology,
Hanoi University of Science and Technology, Hanoi, Vietnam
{sonnh, dungct}@soict.hust.edu.vn

Abstract. An increasingly huge amount of sport news published from a number of heterogeneous sources on the Web brings challenges to the traditional searching method using keywords. Providing an expressive way to retrieve news items and exploiting the advantages of semantic search technique in the development of Web-based sports news aggregation system is within our consideration. This paper presents a method to translate natural language questions into queries in SPARQL, the standard query language recommended by W3C for semantic data. Our contribution consists mainly of the construction of a semantic model representing a question, the detection of ontology vocabularies and knowledge base elements in question, and their mapping to generate a query. We evaluate the method based on a set of natural language questions and the results show that the proposed method achieves good performance with respect to precision.

Keywords: Natural language interface · Semantic web · Question answering
Sport news aggregator · SPARQL

1 Introduction

The Web is considered one of the most popular sports news sources that is quickly updated about worldwide events but the readers encounter an enormous amount of news items including coincident and redundant information or information which is outside their interest area. Therefore, the rapid, convenient and effective retrieval of information for readers is always a challenging issue with regard to the development of Web-based news systems [9]. However, keyword search often returns news that contains the terms in questions, but without understanding the meaning of the desired questions. The semantic search promises to solve above problems by returning results relevant to the readers' desire. Furthermore, besides reading news, readers are also interested in information related to entities appearing in news, such as certain characters or organizations. This makes features for sport news. For instance, readers are likely to view news about Lionel Messi, Christiano Ronaldo together with "the classic matches".

T. Q. Duong and N.-S. Vo (Eds.): INISCOM 2018, LNICST 257, pp. 63–73, 2019.
https://doi.org/10.1007/978-3-030-05873-9_6

In previous work, we proposed a sports news system based on semantics and discussed benefits that the system could bring and introduced methods for generating RDF semantic annotation for news items [7, 8]. Our next research topic involves the building of a news semantic search system which can return both precise and suitable news in accordance with readers' requirement. Moreover, this search system should be friendly to users who are just normal readers with limited knowledge about technology. In this paper, we present a method for translating questions from natural language form, for example "*Which team defeated Chelsea this season?" or "Who transferred to Barcelona this year?"* into SPARQL queries. These queries will be sent to semantic search engine [4] to find related news associated with the answers to questions.

The remainder of this paper is organized as follows. We start with the presentation of some existing works on information retrieval, especially the systems support natural language questions in searching for information. Section 3 describes the method for translating a natural language question in sport domain to SPARQL query. Experimental scenario and results are presented in Sect. 4, followed by the conclusion.

2 Related Works

There were many researches on information retrieval from semantic data store, and some of researches using directly SPARQL statements to query information from data of semantic knowledge store [12]. However, the use of SPARQL syntax has some weaknesses such as complex query language syntax. In addition, it requires users to understand inside structure of semantic knowledge store. Several other researches enhance the friendliness to users by providing graphical user interface based on ontology to formulate SPARQL queries [2]. Although these graphical interfaces make the query formulation work easier, they are still not perfectly suited to end-users as they still requires users to basically understand ontology.

Providing an intuitive way to express complicated user needs, natural language interfaces such as question answering (QA) systems have received increasing attention. Guo and Zhang [6] raised the importance and feasibility of a Question-Answering system in the Chinese language. Their QA system was established relying on three models, i.e. the question's semantic comprehension model based on Ontology and Semantic Web, FAQ-based question similarity matching model, and document warehouse-based automatic answer fetching model. Certain works based on controlled natural languages, such as Squall2Sparql [5] typically consider an unambiguous restricted subset of natural language that can be translated directly into SPARQL.

PANTO [11] is a Portable nAtural language interface to ontologies allowing users to represent "information needs" by means of natural language without knowledge of RDF, OWL syntax, query languages, or vocabularies of the ontologies. It uses WordNet and String metric algorithm to map words in the user's query to entities in the ontology (concepts, entities, relations). However, the system still has limitations in processing negative questions, and is unable to deal with the question of quantity.

QuestIO [3] (Question-based Interface to Ontologies) is a domain-independent tool which uses natural language to serve query of large knowledge stores in ontology. The drawback of this tool comes from the detection of relations in the question as it is based

on rules and not on syntax analysis of questions at deep level. Therefore, it cannot deal with questions with complex semantic. The test was carried out on data set of 22 questions from GATE user mailing list, where users enquired about various GATE modules and plug-ins, the precision obtained was 71.88%.

ORAKEL [1] is a nature language interface which accepts questions and returns answers on the basis of a given ontology. This task is performed thanks to the Query Interpreter, which interprets the input question and transforms it to a first order logic representation and the Query Converter, which is in charge of transforming the logical representation of the question to SPARQL query. This system can only work with the type wh-question type and not the yes/no question type.

The above works show the significance of semantic search function in the form of natural language question. However, they focus more on general domain than specific one. In addition, porting Natural Language Interfaces to other domains [1] is never an easy task. The question types are identified by above systems usually having simple structures, which cannot describe all of the readers' demand of information.

Our research aims to develop a search system using natural language, friendly to users, allowing users without knowledge about complex query language to use the system effectively. In order to do so, the precision of transforming query into its semantic form needs to be enhanced. We propose a novel method consisting of several stages in order to do this in the domain of sports news. Question modeling, analysis of grammar structure recognition and transforming to corresponding semantic representation play an important role in our method.

3 Translating Natural Language Question into SPARQL

The proposed process of translating natural language question to SPARQL query comprises 5 main phases as presented in Fig. 1. The procedure is as follows. The input question in the form of natural language is normalized by the Question Preprocessing component, which helps for others components to function effectively and precisely. After that, the question is sent to Syntax Analysis to analyze grammar elements and its relations. Then the question is represented in the form of semantic model by the component of Semantic Representation of Question. From the semantic model, the

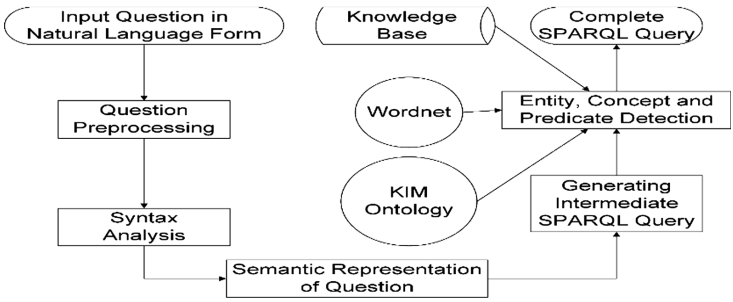

Fig. 1. Natural language to SPARQL query translation process

intermediate SPARQL query is generated. Component of named entity, concept and predicate detection then annotates variables in the intermediate SPARQL query by URIs in ontology and knowledge base of the system. And at last, a complete SPARQL query is generated.

3.1 Question Preprocessing

Preprocessing component is responsible for normalizing abnormal tokens and determining the temporal property of question. Readers are accustomed to use abbreviations when they write. Therefore, we make statistics of common abbreviations and build a normalization table. Each token in questions is examined, and then abbreviations are replaced. In parallel, statistics of temporal labels were classified into the following types: one-day period (i.e. "today", "yesterday", etc.), one-week, one-month and one-year period. Based on the time readers make a query, a specific temporal value corresponding to each temporal label is computed, then replaces temporal label in questions with this specific value.

The system accepts even input as imperative sentences and reduced ones. In order for syntax analyzer to function properly and to simplify next steps processing, the system transforms such sentences into one of two standard question forms with equivalent meaning. The former is wh-question and the latter is yes/no question. For example, the question *"news about Lionel Messi"* is transformed into better standard grammar question as *"Which news is about Lionel Messi?"*.

3.2 Syntax Analysis

Syntax Analysis is in charge of identifying the form of question, grammar elements and its relation in questions, thanks to the POSTagged, Phrase Structure Tree and Typed Dependencies analysis. The results are used in tasks of next phases such as identification of question form, building triple relations, annotation of entity, class and property.

Phrase Structure Tree is a visual way to represent the output of the syntax analysis process. Phrase Structure Tree shows 3 aspects of sentence structure: linear order of words in the sentence, groups of words to establish a phrase and hierarchical structure of the phrase. There, the root node of tree identifies form of question. The Typed Dependencies represent the grammatical relations between words in a sentence. Each Typed Dependency is a triplet: name of the relation, governor and dependent.

For example, for the sentence "Which news is about Lionel Messi?", the system figures out typed dependencies as follows: `det(news-2, Which-1), attr(is-3, news-2), root(ROOT-0, is-3), nn(Messi-6, Lionel-5), prep_about (is-3, Messi-6)`. Where, Det (determiner) is the relation between the head of a NP and its determiner; Attr (attributive) is a relation intended for the complement of a copular verb such as "to be", "to seem", "to appear"; Root: the root grammatical relation points to the root of the sentence; Nn: is a noun compound modifier relation; Prep: A prepositional modifier of a verb, adjective, or noun is a prepositional phrase that serves to modify the meaning of the verb, adjective, noun, or even another preposition.

In this work, only certain Typed Dependencies are concerned, which show constraints between elements of question, such as subject-verb, verb-object, passive verb-agent, word-accompanied preposition, noun-adjective modifier, etc. They help to identify important words in the sentence and relations between them. On that basis, constraints by triple patterns are formulated in the SPARQL query.

3.3 Building Semantic Representation of Question

We define a semantic representation model covering two basic question forms: wh-question and yes/no question as the aggregation of the following elements: variable list, constraints of variables and constraints of dependent relations. This model plays the role of the intermediate representation to generate SPARQL query.

Each variable in the *Variable list* represents a token in the question and is named in the format "symbol string + ID (for example: `?x1`, `?x2`, etc.). Variable label is the word it represents. Variables are divided into two types, in which *query variables* are hidden variables containing information to be queried for the question and *ordinary variables* are the rests. For wh-question, at least one query variable is required to exist in the variable list while for yes/no question, query variable does not exist.

Two types of query variable are proposed, quantity query variable with wh-question of "how many", represented in variables list as `COUNT(?variable_name)` and object query variable with wh-question of "who/what/which/where", represented in variables list as `?variable_name`.

Constraints of variables include the constraint on the label of variables, constraints on relationship between variables and quantity constraints. We consider the quantity constraint of certain variable by a specific value through relationships as: "=" (equal), ">" (morethan), "> = (moreOREqual)", "<" (lessthan), "<=" (lessOREqual).

Constraints of dependent relationship include AND/OR constraints indicating the dependent relationships occurring at any time and temporal constraints.

Transforming Grammar Structure into Semantic Representation

This task consists of four main steps: identification of query variable, identification of ordinary variables and constraint of dependent relationship between variables and identification of quantity and temporal constraints.

Identification of Query Variable

The existence of a query variable in the variable list depends on the form of input questions. If the input is a yes/no question, query variable does not exist in the variables list. On the contrary, if the input is a wh-question, then subject is identified corresponding to query words. For "who/what/where", we determine subject as the query word, while with "how many/which", it is determined as the noun following a query word, by which the query variable is identified.

Identification of Ordinary Variables and Constraint of Dependent Relationship Between Variables

Each Typed Dependency is a triplet, i.e. name of the relation, governor and dependent. From Typed Dependencies, related words and relations among them (name of Typed

Dependency) are inferred mutually. These words are represented by variables, i.e. query variables and ordinary variables).

An important part of a SPARQL query is triple set what defines the constraints of the question. However, each dependent relationship between variables which are identified from Typed Dependency is just in the form of double. These relationships belong to one of two types, subject - predicate or predicate – object. Therefore, we propose to combine typed dependencies to generate constraint by triple patterns. Two typed dependencies representing two types of different dependence which share predicate are to determine to create the triple relationship in the form (subject, predicate, object). The following examples illustrate how to carry out the method for questions with simple semantics S-V-O. For the question "Did Barcelona defeat Chelsea?", the syntax analysis phase finds out two main typed dependencies which are `nsubject(?x-defeat, ?y-Barcelona)` and `dobject(?x-defeat, ?z-Chelsea)`. Where, `nsubject` represents the subject – predicate relationship and `dobject` represents the predicate – object relationship. By combining these two typed dependencies, we obtain a triple pattern `(?y-Barcelona, ?x-defeat, ?z-Chelsea)`.

Identification of Quantity Constraint
Two types of quantity constraints are considered:

(1) Constraint based on comparing the number of certain objects with a specific value, i.e. "Who scored more than 3 goals?" and
(2) Superlative quantity constraint of a certain object, i.e. "Who scored the most goals?" "Which team conceded the least goals?"

For the type (1), the type dependency `num(?object, ?quantvalue)` shows the existence of a quantity constraint for the `?object` object based on the relationship with the `?quantvalue` value. In order to identify the relationship between an object and this quantity value, we consider the existence of another typed dependency which is `quantmod(?quantvalue, "than")`.

The detection of "the most/least" + noun structure can be based on two typed dependencies `det(?object, "the")` and `amod(?object, ?dep)`. For example, if the value of `?dep` in the typed dependency `admod` is "most", it means the `?object` object has the most occurrences, the module generates `themost(?object)` constraint in the semantic model.

Identification of Temporal Constraint
A raised problem is to identify elements of time-related question and to transform it into a semantic model. To do that, in the semantic model, we define one "Interval" including two fields: `Interval (BEGIN, END)`. Interval type shows the constraint that time of events must be in the form of from BEGIN to END.

In the scope of this work, we only concern minimum time unit in day (day, week, month, year). If the question mentions temporal context, then the syntax analyzer generates Typed Dependency `prep_in (object, time_label)`. In which, `time_label` is the temporal label to show the temporal context of question. If `time_label` is a certain day, BEGIN and END attain the same value. If `time_label` is a certain week, month, year or season, we base on time user making queries to compute BEGIN and END value.

3.4 Generation of Intermediate SPARQL Query

The SPARQL intermediate queries just have frames containing variables, which consist of two main elements, namely question clause and condition clause. In addition, for special questions such as questions with quantity constraint and question with temporal constraint, there are additional constraint clauses. Figure 2 shows the process of generating these queries from semantic model of question and main principle of each step as well.

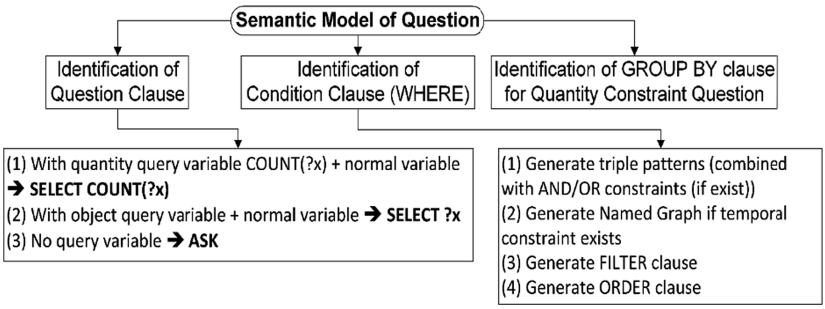

Fig. 2. Generation of intermediate SPARQL query.

3.5 Entity, Class and Predicate Detection and Complete SPARQL Generating

Type and value for these variables in the semantic representation model are identified in the phase of ontology's elements detection. Each label of variable is mapped to knowledge base and ontology to detect corresponding entity, class or property.

Named Entity Detection

KIM [10] is a platform that provides a knowledge and information management infrastructure, services for automatic semantic annotation, indexing and document extraction. However, KIM is built to serve in a public domain and it is equipped with a high level ontology named PROTON and a knowledge base with a large number of entities of general importance. In order to identify name entities in news item, we extend KIM for serving a specific domain such as sport. An ontology named BKSport with classes and properties at a lower and more detailed level is integrated into KIM' ontology, as shown in Fig. 3.

By enriching the KIM's knowledge base, named entities appearing in input questions are detected as instance of detail class in sport ontology rather than the one of KIM's general types. Each variable in the question's semantic model has a corresponding private label which is used to semantically annotate variables. Most of variables have been labeled with the name of proper noun corresponding to an entity in the knowledge base. If a variable is recognized as an entity in the knowledge base, it is replaced with a URI of the entity. In case that a label is proper noun but not recognized, a FILTER statement is added into intermediate SPARQL query to bind label value to variable.

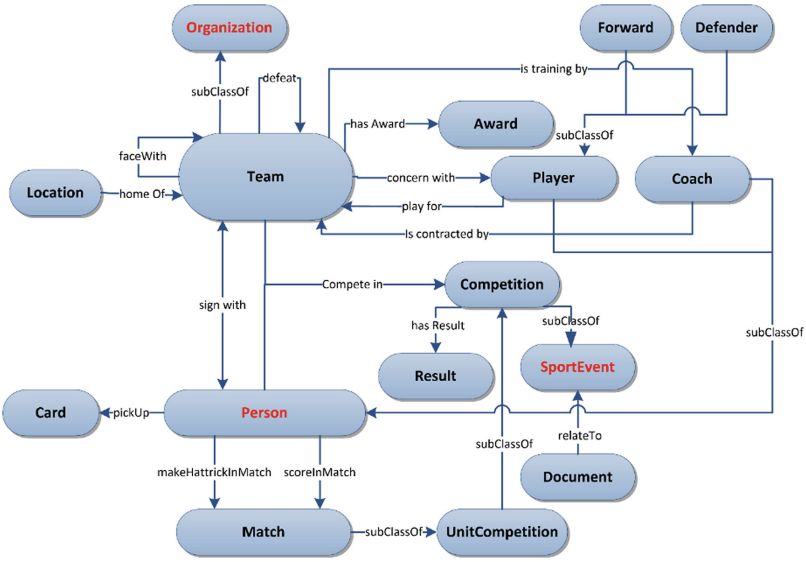

Fig. 3. Snapshot of a part of the BKSport ontology

Classes and Properties Detection

For variables act as subject or object in the constraint by triple patterns but not entities, class constraint for such variables is built. To recognize class of variables, firstly a list with two fields is built. The former is URIs of all classes in ontology and the latter is corresponding labels of those classes. Secondly, WordNet is applied to search synonyms of labels of each above URI, generating a set of representing words for each URI. Finally, the system examines which set of representing words the labels of each variable belong to. From that, we identifies corresponding URIs to variables and adds a triple pattern with the syntax of `<variable_name> <rdf:type> <class_URI>` into SPARQL query to identify type of variables.

Property recognition of variables acting as predicate in triples is also carried out like class recognition.

Generation of Complete SPARQL Query

As results of the detection phase, all variables in semantic model are identified. Generation of complete SPARQL query is simply accomplished by replacing variables in the intermediate SPARQL query by corresponding URIs. For example: a SPARQL query is generated as follows for the question "Which team defeated Chelsea in 08/08/2015?":

```
SELECT ?x WHERE {
?graph {  ?x  <http://bk.sport.owl#defeat> <http://bk.sport.owl#Chelsea>.      }
?x rdf:type <http://bk.sport.owl#team>.
?g <http://bk.sport.owl#hasTime> ?t.
?t rdf:type time:Instant.
?t time:inXSDDateTime ?instantDate.
FILTER (?instantDate >= "2015-05-08"^^<xsd:dateTime> && ?instantDate <= "2015-05-08"^^<xsd:dateTime>).
```

4 Experiments

To evaluate the proposed method, we setup an experiment dataset consisting of 41 questions belong to frequently used question in sports domain. These questions are sent to the system to be transformed into SPARQL queries automatically. Evaluators then consider whether the generated SPARQL query describes information shown in the natural language input question accurately and fully or not.

However, "correct level" evaluation of SPARQL query is difficult. We determine a SPARQL query including three main clause types including *question clause, WHERE clause*, *other constraint clauses* (for example temporal constraint clause, quantity constraint clause, etc.). In order to measure the precision of query, we firstly measure the precision of each type of clauses. In order to do it, we base on unit elements. We define a "correct unit element" which is a variable satisfying one of the following conditions: *recognized, corresponding to one URI, type is identified clearly and constraints on label value are identified clearly.*

We evaluate precision of a query generated by the system based on not only satisfied number of elements but also importance of each element. Weight is assigned for each type of clause in query based on our opinion about its importance. Considering n_i as the number of correct unit elements of clause i, N_i is the number of unit elements of clause i needed to identify in a query written by an expert, then $\frac{n_i}{N_i}$ is the precision of clause i. In special case, there is no specific question variable in "SELECT *" question clause, we assign 0.5 to the precision of question clause.

Finally, we determine the general formula to measure the precision of a query q generated by the system as follows:

$$Precision(q) = b \times \frac{\sum_{i=1}^{M} \left(a_i * w_i * \frac{n_i}{N_i} \right)}{\sum_{i=1}^{M} a_i w_i}$$

In which:

- b gets the value of 0 or 1:
  - $b = 0$, if identification of question clause (SELECT or ASK) is wrong, or wrong identification of all question variables.
  - $b = 1$ in the other cases.
- M is the number of clause types in the query written by experts.
  - $a_i = 1$ if the clause exists in the query generated by the system, $a_i = 0$ otherwise.

Table 1 illustrates a part of the experiment dataset of questions and the corresponding precision computed using proposed formula. For the whole set of questions, the average formula in measurement of all experiment questions is applied and the overall precision obtained is 91.89%.

Table 1. Certain input questions and corresponding precision of transforming process

ID	Question	Precision
	*** *Definition question*	
1	Who is Lionel Messi?	1
	*** Yes/no question	
2	Was Chelsea defeated by Barcelona last year?	1
3	Did Barcelona defeat Chelsea?	1
4	Did Wayne Rooney dispute with Alex Ferguson yesterday?	1
	*** *Predicative question*	
5	Which team defeated Chelsea?	1
6	Which team defeated Chelsea this season?	0.83
7	Which event relates to Lionel Messi?	1
8	Which team did Lionel Messi transfer to?	1
	*** *Opinion question*	
12	What did Lionel Messi say about Chelsea?	1
	*** Phrase-verb	
13	News about Chelsea	1
	*** *Quantity question*	
14	How many clubs defeated Chelsea?	1
	*** Comparative, superlative question	
15	Who won the most games this year?	1
16	Who won more than 1 game this year?	1
	*** *Association question*	
20	What is the result of the match between Chelsea and Barcelona?	1
21	What happened between Chelsea and Barcelona?	1
	*** *Multi-subject, multi-object question*	
22	Which team defeated Chelsea and Barcelona?	1
23	Did Manchester United and Chelsea defeat Barcelona in 2014?	1
24	Was Barcelona defeated by Manchester United and Chelsea this year?	1

5 Conclusion

We have presented a method of transforming natural language query into SPARQL query. Based on grammar structure preprocessing and deep analysis of questions, the proposed system is able to process some complex forms of questions such as comparative question, superlative question, question with many subjects and objects, question with abnormal grammar structure, etc. Although the proposed method handles improperly in some cases due to the complexity of generated syntax analysis results or the complexity of question semantics, such complex questions were handled partly correctly. By experiment and evaluation with question set including various questions, the proposed system obtains high precision and competitive with related approach. In future works, we focus on improving cases that the current system cannot handle properly. “The tense” of relationship which is verb should be studied to capture semantics of question more correctly.

References

1. Cimiano, P., Haase P., Heizmann, J.: Porting natural language interfaces between domains: an experimental user study with the orakel system. In: Proceedings of the 12th International Conference on Intelligent User Interfaces, pp. 180–189 (2007)
2. Clemmer, A., Davies, S.: Smeagol: a "specific-to-general" semantic web query interface paradigm for novices. In: Hameurlain, A., Liddle, Stephen W., Schewe, K.-D., Zhou, X. (eds.) DEXA 2011. LNCS, vol. 6860, pp. 288–302. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23088-2_21
3. Damljanovic, D., Tablan, V., Bontcheva, K.: A Text-based query interface to OWL ontologies. In: Proceedings of the 6th Language Resources and Evaluation Conference (LREC), Morocco, ELRA, pp. 205–212 (2008)
4. Erling, O., Mikhailov, I.: RDF support in the virtuoso DBMS. In: Pellegrini, T., Auer, S., Tochtermann, K., Schaffert, S. (eds.) Networked Knowledge—Networked Media, vol. 221, pp. 7–24. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02184-8_2
5. Ferré, S.: SQUALL: a controlled natural language for querying and updating RDF graphs. In: Kuhn, T., Fuchs, N.E. (eds.) CNL 2012. LNCS (LNAI), vol. 7427, pp. 11–25. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32612-7_2
6. Guo, Q., Zhang, M.: Question answering system based on ontology and semantic web. In: Wang, G., Li, T., Grzymala-Busse, J.W., Miao, D., Skowron, A., Yao, Y. (eds.) RSKT 2008. LNCS (LNAI), vol. 5009, pp. 652–659. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-79721-0_87
7. Nguyen, Q.M., Cao, T.D.: A novel approach for automatic extraction of semantic data about football transfer in sport news. J. Pervasive Comput. Commun. **11**(2), 233–252 (2015)
8. Nguyen, Q.M., Cao, T.D., Nguyen, H.C., Hagino, T.: Towards efficient sport data integration through semantic annotation. In: Proceeding of The Fourth International Conference on Knowledge and Systems Engineering, KSE 2012, pp. 99–106 (2012)
9. Paliouras, G., Mouzakidis, A., Moustakas, V., Skourlas, C.: PNS: a personalized news aggregator on the web. In: Virvou, M., Jain, L.C. (eds.) Intelligent Interactive Systems in Knowledge-Based Environments, vol. 104, pp. 175–197. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-77471-6_10
10. Popov, B., Kirayakov, A., Ognyanoff, D., Manov, D., Kirilov, A.: KIM—a semantic platform for information extraction and retrieval. Nat. Lang. Eng. **10**(3/4), 375–392 (2004)
11. Wang, C., Xiong, M., Zhou, Q., Yu, Y.: PANTO: a portable natural language interface to ontologies. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519, pp. 473–487. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72667-8_34
12. Yamaguchi, A., Kozaki, K., Lenz, K., Wu, H., Kobayashi, N.: An intelligent SPARQL query builder for exploration of various life-science databases. In: Proceedings of the 3rd International Conference on Intelligent Exploration of Semantic Data, pp. 83–94 (2014)