Identification of the Question Focus: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction

Danica Damljanovic, Milan Agatonovic, Hamish Cunningham

Department of Computer Science
University of Sheffield
Regent Court, 211 Portobello Street
S1 4DP, Sheffield, UK
{d.damljanovic,m.agatonovic, h.cunningham}@dcs.shef.ac.uk

Abstract

Most question-answering systems contain a classifier module which determines a question category, based on which each question is assigned an *answer type*. However, setting up syntactic patterns for this classification is a big challenge. In addition, in the case of ontology-based systems, the answer type should be aligned to the queried knowledge structure. In this paper, we present an approach for determining the answer type semi-automatically. We first identify the *question focus* using syntactic parsing, and then try to identify the answer type by combining the head of the focus with the ontology-based lookup. When this combination is not enough to make conclusions automatically, the user is engaged into a dialog in order to resolve the answer type. User selections are saved and used for training the system in order to improve its performance over time. Further on, the answer type is used to show the feedback and the concise answer to the user. Our approach is evaluated using 250 questions from the Mooney Geoquery dataset.

1. Introduction

Research has been very active in last decades in designing Natural Language Interfaces (NLIs) to various representations of the data among which most serve as an interface to relational databases, e.g., (Popescu et al., 2003), (Thompson et al., 2005), (Hallett et al., 2007), and many others, or ontologies, e.g., (Cimiano et al., 2007), (Lopez et al., 2007), (Wang et al., 2007), (Damljanovic et al., 2008). The major difference of these in comparison to traditional open-domain question answering systems is the underlying knowledge representation - whereas in open-domain question-answering (QA) systems the answer is typically derived from unstructured data (e.g. documents), NLIs to structured data allow users to interact with a system using written or spoken language (e.g., English) to perform tasks that usually require knowledge of a formal query language (such as SQL or SPARQL). Intention behind building NLIs to structured data is enabling the users with no knowledge of formal languages to use them with minimal, ideally, no training. These systems are often referred to as *closed-domain question answering* systems. All these systems have Natural Language question as input and the major difference is the underlying structure of the knowledge which contains the answer.

Most QA systems contain a classifier module which detects a question category, based on which, each question is assigned an *answer type*. However, setting up syntactic patterns for this classification is not trivial (Ferret et al., 2001). These are usually derived from the dataset (e.g. corpora), which must be large in order to work efficiently (Ferret et al., 2001). Moreover, classical question-answering approaches do not often apply to all domains. For example, in (Niu et al., 2003) the differences between general and medical QA are outlined.

In ontology-based systems, the answer type is usually aligned with the queried knowledge structure, as this could change over time (for example, if the ontology which is being queried changes or if the system is being ported to work with a completely different domain/ontology). It is not trivial to translate an arbitrary question into a relevant logical representation or a formal query which will lead to the correct answer (Mooney, 2001).

In this paper, we present an approach for determining the answer type semi-automatically. We first identify the *question focus* using syntactic parsing, and then try to identify the answer type by combining the head of the focus with ontology-based lookup. When this combination is not enough to make conclusions automatically, the user is engaged into a dialog in order to resolve the answer type. Further we present how we use the answer type to show feedback and the concise answer to the user.

We evaluate our approach using the Mooney Geoquery dataset[1].

This paper is structured as follows. In Section 2. we detail the algorithm for identification of the answer type. In Section 3. we show how we use the identified answer type for presentation of the concise answer and feedback within our Natural Language Interface to Ontologies called FREyA. In Section 4. we present evaluation results with the Mooney Geoquery dataset, and finally we conclude in Section 6.

2. Identification of the Answer Type

Most QA systems classify questions based on the type such as *What*, *Why*, *Who*, *How*, *Where*, which is followed by the identification of the answer type. *Answer type* refers to the type of the answer, such as *Person* or *Organisation* for questions starting with *Who*. However, identifying the answer type is not always sufficient for finding answers as it might not say much about the query itself. Therefore, in (Moldovan and Harabagiu, 2000) the identification of the answer type is followed by the identification of the focus. According to (Moldovan and Harabagiu, 2000), a *focus* is a word or a sequence of words which define the question and

[1]http://www.cs.utexas.edu/users/ml/nldata.html

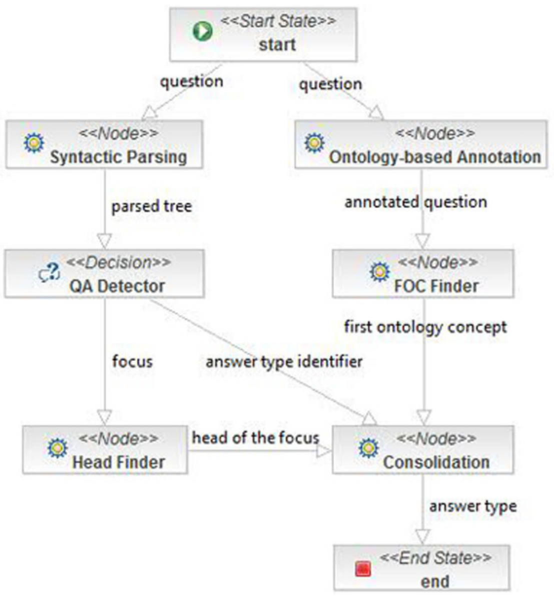


Figure 1: Workflow for the identification of the answer type

disambiguate it by indicating what the question is looking for. For example, in *what is the largest city in Germany?* the focus is *largest city*. Unlike their approach which is inline with traditional approaches used in open-domain QA systems, we skip the identification of the question category, and first try to identify the focus of the question, which is used in the subsequent steps to identify the answer type.
Figure 1 shows the workflow for the identification of the answer type. *QA Detector* combines the syntactic parsing with a set of the heuristic rules in order to find the focus. For the specific types of questions, the focus is not so important for identification of the answer type, and these questions usually have the *Answer Type Identifier (ATI)*. For example, while the focus in *How long is Mississippi?* is *Mississippi*, what we need to know in order to find the answer type is what *How long* refers to. Therefore, *QA Detector* would find that *How long* is the ATI. During *consolidation* *How long* would be used together with the first ontology concept (*geo:Mississippi*) to generate suggestions for the user. The user's selection would be then saved as the answer type.

2.1. QA Detector

QA Detector combines the output of the Stanford Parser (Klein and Manning, 2002) which generates a syntax tree, with several heuristic rules and here we outline the high level pseudo code:

```
1. find X: the first prepreterminal
2. if X is NP or NN* then focus = X
3. if X is WHADVP || WHNP || WHADJP
   || ADJP
   - if the first child is WRB,
     and the second is JJ or ADJP,
     Answer Type Identifier (ATI) = X
   - if there is only WRB then ATI=WRB
```

A node is a *prepreterminal* if all the children of this node are preterminals. *Preterminal* is defined to be a node with one child which is itself a leaf.
The output of this algorithm could be:

- The *focus* of the question.
- *Answer Type Identifier (ATI)* indicates that additional input is required from the user in order to assign the answer type to the question. ATI string is used to model a dialog together with the underlying knowledge structure in order to generate meaningful suggestions.
- *No match found*: the parser fails to find both the focus and the ATI.

2.2. FOC Finder

For generating ontology-based annotations, we use the GATE (Cunningham, 2005) application based on the OntoRoot Gazetteer (Damljanovic et al., 2008). This gazetteer is available as a plugin of the open-source project GATE (`http://gate.ac.uk`).
FOC Finder identifies the First Ontology Concept (FOC) in the question which is of the `class` or the `datatype property` type. This is because the answer type eventually refers to one of these two types of concepts in the ontology. Therefore, if the FOC refers to other types of ontology concepts, the procedure is as follows:

- If the FOC refers to an `object property`: perform the consolidation with the *domain* or *range classes* of this property
- If the FOC refers to an `instance`: perform the consolidation with a *class* of that *instance*

2.3. Consolidation

For each query the goal is to identify the answer type. Consolidation is an attempt to achieve this by merging the output of the *QA Detector/HeadFinder* with the FOC. While the focus itself is important to capture relevant information which helps in finding the correct answer, the *head of the focus* is what we use in order to find the answer type. We identify the head of the focus using the *ModCollinsHeadFinder* class of the Stanford Parser package, which is a variant of the *HeadFinder* described in (Collins, 1999).
The consolidation algorithm can be described using the following pseudo code:

```
if ATI!=null
   generate suggestions for the user
   the Answer Type = the user's
   selection
else
   if the head of the focus != null
   consolidate it with the FOC in order
   to identify the Answer Type
```

Depending on the relation between the *head of the focus* and the *FOC*, we apply different rules in order to identify the answer type. Both head of the focus and the FOC refer to a word or a set of words in the question. Therefore, they can either overlap, or be placed one before/after another. In the case when these two overlap the consolidation is performed as follows:

- *Exact match*: both the *head of the focus* and the FOC refer to the exactly same word(s) in the question. Therefore, the FOC becomes the *Answer Type* of the question. For example, in *What is the capital of Texas? capital* is the head of the focus (as *capital* is the head of *the capital*). The exactly same string (*capital*) is annotated as the FOC referring to *geo:Capital* in the ontology. As these two overlap meaning that their start and end offsets are equal, the answer type of the question is *geo:Capital.*

- *The FOC is contained within the head of the focus, and vice versa*: the user is asked to decide whether the identified head of the focus refers to the FOC or not.

When the head of the focus and the FOC do not overlap, the consolidation is performed as follows:

- *The head of the focus is before the FOC*: for example, in *what is the area of Idaho?* the focus is *the area*, and the *answer type* cannot be resolved without the dialog; the user must choose that *area* refers to one of the suggestions generated based on the neighbouring ontology-based annotations in the question; this is described in more details in Section 2.4.

- *The system failed to identify the focus or the ATI in the previous step*: in this case FOC becomes the answer type. For example, in *what is the most populous state?* the focus is not identified due to our algorithm relying on *prepreterminals*; in this case, as the FOC refers to *geo:State*, *geo:State* becomes the *answer type*.

- *The head of the focus is after the FOC*: in this case, the FOC becomes the answer type. For example in *what state borders Michigan? borders* is incorrectly identified as the focus while *state* is annotated as the FOC; therefore, the answer type is consolidated into *geo:State*. This consolidation rule is usually used to correct the parser mistakes. Another example is, if the parser identifies the *focus* to be *Nevada* in *which rivers flow through Nevada?*, this is obviously incorrect. During the consolidation phase, if *rivers* is identified as the FOC which refers to *geo:River*, this will cause ignoring the identified focus, and the answer type would be *geo:River*.

2.4. Generating suggestions

A list of suggestions is created based on the ontology reasoning rules, and ranked using combination of the synonym detection and the string similarity (see Section 2.4.1.). For example, in the case of *How big is Alaska?*, where Alaska is recognised as an instance of a type *geo:Country* in the ontology, these suggestions could be a set of datatype properties related to Alaska, or *geo:Country* such as: *geo:stateArea*, *geo:statePopulation*, and the like. Table 1 shows several examples of the identified ATI, suggestions generated based on the FOC in the question, and the answer type as identified after the user's selection. Table 2 shows the answer type identified for the questions which did not have any ATI.

While during the consolidation phase described in the previous section, we give priority to the ontology concepts, particularly to the First Ontology Concept, when a conflict occurs in the case of ATI and ontology-based annotations, the latter is usually ignored. One example is *How long is Mississippi?*. Our algorithm would identify ATI to refer to *How long*. On the other hand, *long* would be annotated as referring to the mountain *Longs* in the ontology. The reason is that the name *longs* is lowercased in the ontology, and therefore, our gazetteer which matches the root of the question term with the root from the lexicalisation available in the ontology, matches *long* in *How long* with the root of *longs*. While we usually give priority to all ontology concepts in comparison to syntactic parse result, this case is an exception: we ignore the FOC and generate suggestions for the user using the neighbouring ontology concept, which in this case happens to be *geo:mississippi*. We then ask the user if *how long* is related to any of the generated suggestions, which would include *geo:length* among others, which is the correct one.

One special case of generating suggestions is when no ontology annotations are found in the question (FOC=null). In this case, we try to generate suggestions by showing the most generic concepts to the user such as top classes and properties.

2.4.1. Initial Ranking

Initial ranking is based on the string similarity between the focus and suggestions, and also based on the synonym detection as identified by Wordnet (Fellbaum, 1998) and Cyc[2]. For the string similarity we combine the Monge Elkan[3] metrics with the Soundex[4] algorithm. When comparing the two strings the former gives a very high score to those which are exact parts of the other. For example, if we compare *population* with *city population*, the similarity would be maximised as the former is contained in the latter. The intuition behind this is the way ontology concepts are usually named. The Soundex algorithm compensates for any spelling mistakes that the user makes - this algorithm gives a very high similarity to the two words which are spelled differently but would be pronounced similarly.

2.5. Learning

Learning is used to improve the ranking of suggestions over time. Each suggestion has its *initial ranking* calculated based on the synonym detection and the string similarity as explained previously. These are used in the untrained system. Each time a suggestion is selected by the user, it receives a reward of +1 while all alternative ones receive -1. The system then learns to place the correct suggestion at the top for any *similar* questions. *Similar* is identified by a combination of the focus/ATI and the FOC. This increases reusability of our learning mechanism as our learn-

[2]`http://sw.opencyc.org/`

[3]see `http://www.dcs.shef.ac.uk/~sam/stringmetrics.html#monge`

[4]`http://en.wikipedia/wiki/Soundex`

Table 1: Sample queries with the identified ATI and the answer type

Query	**Answer Type Identifier**	**First Ontology Concept**	**Suggestions**	**Answer Type**
How big is Alaska?	How big	geo:Alaska (geo:State)	1. geo:stateArea 2. geo:statePopulation 3. geo:isCityOf ... n. none	geo:stateArea or geo:statePopulation
How high is the highest point in America?	How high	geo:hiPoint	1. geo:hiElevation 2. geo:isHighestPointOf 3. geo:hasHighPoint 4. none	geo:hiElevation
Where is the highest point in Hawaii?	Where	geo:isHighestPointOf	1. geo:HiPoint 2. geo:State 3. none	geo:HiPoint

Table 2: Sample queries with the identified focus and the answer type

Query	**Focus**	**First Ontology Concept**	**Suggestions**	**Answer Type**
What rivers run through Colorado?	rivers (head: rivers)	geo:River	-	geo:River
What is the smallest city in Alaska?	the smallest city (head: city)	geo:City	-	geo:City
What is the population of Idaho?	population (head: population)	geo:Idaho (geo:State)	1. geo:statePopulation 2. geo:stateArea 3. none	geo:statePopulation

ing model is not updated per question, but per each combination of the focus/ATI and FOC. In addition, we apply some generalisation rules derived from the ontology. For example, if the FOC is *geo:Capital*, we would save its superclass *geo:City* in our learning model in order to reuse the same rule for all *cities*, not only for *capitals*.

The advantage of our learning mechanism in comparison to the ones which use a set of predefined rules, is that keywords such as *Where*, *When* or *How many* do not have to be manually 'mapped' to some predefined categories. For an unknown knowledge structure, it might be indeed very hard to perform this mapping manually, and our user-system interaction can help in that sense.

More details about the learning algorithm is given in (Damljanovic et al., 2010).

2.6. Example

The result of running the algorithm for the identification of the answer type over *what are the highest points of states bordering Mississippi?* is shown in Figure 2.

Words which are highlighted in red (*states*, *mississippi*) are those which refer to the ontology-based annotations. Red lines (borders) are those which are found based on the ontology reasoning. The blue highlight (*the highest points*) refers to the identified *focus* following our algorithm. As in this example, the identified focus is not related to the ontology annotations, it is used to generate suggestions for the user. The user will be prompt with the dialog which looks like in Figure 3.

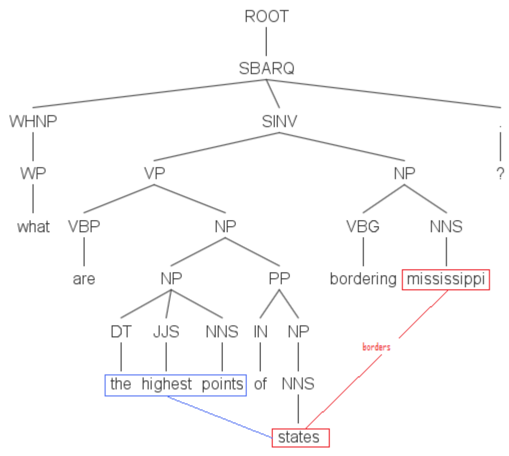
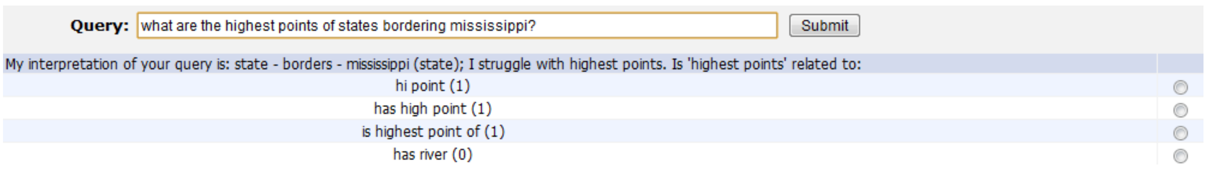


Figure 2: Combining syntactic parse tree with ontology-based annotations

The trade off is that in our initially untrained system, the user will see much more options the ones he is interested in, but the learning mechanism which works behind the scene would put the correct ones at the top by the time.

3. Natural Language Interfaces to Ontologies: presentation of results

In this section we describe how we use the *answer type* identified as described previously, in our Natural Language


Query: what are the highest points of states bordering mississippi?

My interpretation of your query is: state - borders - mississippi (state); I struggle with highest points. Is 'highest points' related to:

hi point (1)	○
has high point (1)	○
is highest point of (1)	○
has river (0)	○

Figure 3: Clarification dialog generated based on the detected answer type

Interface to ontologies called FREyA (see (Damljanovic et al., 2010) for details about FREyA).
Natural Language Interfaces to ontologies translate Natural Language into formal languages such as SPARQL. This translation is what most of existing NLIs focus on, and the problem of showing the results to the user is somewhat de-emphasised. While the most obvious approach is showing the results as they are returned after executing a SPARQL query, a verbose option seems to be preferred by users. Namely, (Kaufmann and Bernstein, 2007) conducted a usability study, which compared four types of query language interfaces to knowledge bases and involved 48 users of general background. Among these, Querix (Kaufmann et al., 2006) was preferred to the others because it returned the answer in a form of a sentence, in contrast to the list of answers returned by the other 3 systems. For example, the question *How many rivers run through Colorado?* was answered by Querix as: *There are 10* (Kaufmann and Bernstein, 2007), while the other 3 systems returned the list of rivers and the number of results found. Because of the way Querix displayed the answer, users had the impression that the system really understood them, and trusted the system more (Kaufmann and Bernstein, 2007).
On the other hand, the survey which compared various NLIs to ontologies and their evaluation results in (Damljanovic and Bontcheva, 2009) shows that the low performance and the error rate is usually caused by:

- Users not being familiar with the domain
- Knowledge is not in the ontology/knowledge base but the system is not capable of guiding the user to rephrase the question
- Feedback messages not helpful i.e. the user can not figure out how to proceed further
- Users have assumptions/misconceptions about the system capabilities and the supported language

This emphasises the importance of showing the system's interpretation to the user, and communicating the message of what the system understood clearly. Therefore, after we identify the answer type, we use it in FREyA to:

- Display the concise answer to the user
- Show feedback

3.1. Display the concise answer

The result of a SPARQL query is a graph, and an important decision to make is how to display results to the user. After the consolidation phase, the answer type is mapped to the ontology concept which could either be: a *class*, or a *datatype property*. Other ontology concepts are resolved to these two before showing results (see Section 2.2.).
Based on the type of the ontology concept, we use different albeit similar patterns for displaying the concise answer:

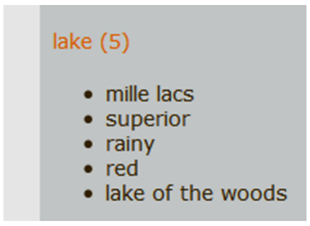
- Answer type is mapped to a *class*: in this case, the answer is usually the list of instances of this class, and the pattern looks like:

  ```
  CLASS (number of answers):
  instance 1
  instance 2
  ...
  instance n
  ```

- Answer type is mapped to a *datatype property*: the answer is the value of this property and the pattern is as follows:

  ```
  DATATYPE PROPERTY (number of values):
  value 1
  value 2
  ...
  value n
  ```

For example, in case of *Show lakes in Minnesota*, *lakes* is identified as referring to *geo:Lake* which is the answer type of the question. As *geo:Lake* is a class we render it as shown in Figure 4.


lake (5)

- mille lacs
- superior
- rainy
- red
- lake of the woods

Figure 4: List showing the answer to the query *Show lakes in Minnesota*

3.2. Feedback

In addition, as feedback can help the user familiarise himself with the queried knowledge structure, we also display *the system's interpretation of the query*: this is visualised as a graph, where we place the answer type in the center, and the answer on the nearest circle, see Figure 5. The user can click on any node in order to investigate it further - each

click will cause the graph to be re-rendered and the clicked node will be placed in the center.
We use JIT library[5] for graph visualisation.

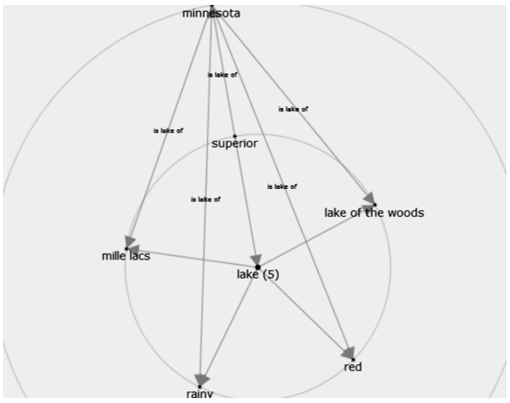


Figure 5: Graph showing the system interpretation of the query *Show lakes in Minnesota*

4. Evaluation

We have evaluated our approach with 250 questions from the Mooney Geoquery dataset.
First we have experimented with the *QA Detector* in isolation, and calculated to which extent it was possible to identify the question focus/ATI using the algorithm described in Section 2.1. This shows the correctness of the *QA Detector* irrespective of whether the answer type was correctly found in the subsequent steps, or not.
The second experiment evaluates the correctness of the consolidation algorithm from Section 2.3. used to identify the answer type.

4.1. QA Detector algorithm

We have first manually labeled the correct focus/ATI for all 250 questions. This was the gold standard for this step.
Out of 250 questions, the ATI was correctly identified for 45 of them (questions starting with *how big*, *how large*, *where* and the like). The results for the remaining 205 were as follows:

- *Correct*: For 174 out of 205 (84.88%), the focus was identified correctly
- *Not found*: For 2 questions (0.97%), our algorithm could not identify neither the focus nor the ATI. This is due to the complex structure in which the prepreterminals were not tagged as noun phrases or nouns. For example, in the questions *what is the most populated state bordering Oklahoma?* or *what is the most populous state?*, the correct focus is *the most populous state* for both questions, however, this noun phrase is not prepreterminal due to *most populous* being tagged as ADJP (see Figure 6).
- *Incorrect*: Remaining 29 (14.15%) questions had incorrectly identified focus and errors could be represented through the following patterns:

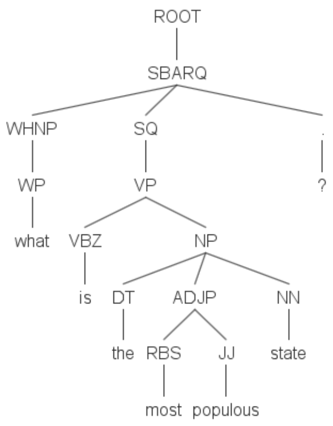


Figure 6: Failure to identify the answer type as no prepreterminals are nouns/noun phrases

  - **Negation**: one sentence with negation had been parsed incorrectly: in *What rivers do not run through Tennessee?*, the parser tagged *rivers* as RB (adverb), while it should be Noun. It is interesting that the same sentence with omitted *not*, is parsed correctly (i.e. *rivers* is tagged as noun (NNS)).
  - **What NP**: such as in *What capital is the largest in the US?* and *What city has the most people?*; while the parser correctly identified the span which contains the focus (*What capital* and *What city* respectively), the head finder identified the head of both phrases to be *What*.
  - **Give me NP**: the possessive pronoun *me* was tagged as PRP which is correct; however, as it was identified as a part of the prepreterminal noun phrase, our algorithm wrongly identified it as the focus; for example, in *Give me the cities in Virginia?* or *Give me the largest state?*, correct focus is *the cities* and *the largest state* respectively, and not possessive pronoun *me* as identified by our algorithm.
  - **State vs. Borders**: When occurring together, these two words have been tagged incorrectly by the parser; for example, in *Which states borders Arkansas?* state is identified as VBZ (Verb, present tense, 3rd person singular) while *borders Arkansas* is NP consisting of NN (borders) and NNS (Arkansas); therefore, the focus is identified to be *border Arkansas*, which is incorrect.

4.2. Consolidation

Further on, we have evaluated the consolidation algorithm in order to identify

- The number of questions for which the focus could be used to successfully identify the answer type with and without engaging the user
- The number of questions for which the answer type was identified correctly although the focus was incorrectly identified in the previous step

[5]www.thejit.org

	Answer type		
Focus	**Required 1 dialog with the user**	**Automatically consolidated**	**Incorrectly consolidated**
correct (174)	65	106	3
not found (2)	0	2	0
incorrect (29)	4	25	0
ATI found (45)	45	0	0
total (250)	114	133	3

Table 3: Results of identifying the answer type using the consolidation algorithm

The results shown in Table 3 show that all questions which had the focus identified incorrectly in the previous step, had the answer type identified correctly after the consolidation phase. However, 4 out of 29 questions involved the user into the dialog in order to place this mapping.
With regards to 174 questions for which the correct focus was found in the previous step, 106 (60.92%) could be mapped to an ontology concept automatically. 65 (37.36%) questions required the dialog with the user in order to map the answer type correctly, 6 out of which did not have any FOC identified, and were answered by modeling suggestions as explained in Section 2.4.
3 (1.72%) questions had wrongly identified answer type after the consolidation. This was the case for *compound-nominal expressions* which contain several nouns, each of which being annotated as referring to an ontology concept. For example, the phrase *state capital* refers to *geo:Capital* in *what is the largest state capital in population?*. However, both *state* and *capital* are annotated as refering to different ontology concepts (*geo:State* and *geo:Capital*), and our algorithm would give priority to *state* as the first ontology concept in the question. In future, we will consider giving priority to the ontology concepts which are the exact matches with the identified head of the focus, such as in this case.
While identification of the answer type as described in this paper can be seen as overload for the user, our intention is to see whether our learning mechanism can reduce this overload by the time. In addition, by engaging the user into a dialog, he has the full control of the system interpretations and therefore can *train* it towards a very good performance even in cases when the ontology (or a set ontologies which are being queried) does not have human understandable lexicalisations. The evaluation of our learning algorithm is detailed in (Damljanovic et al., 2010), where for 103 questions from the Mooney dataset, our learning mechanism improved the initial ranking of suggestions by 6%.

5. Related Work

Identification of the answer type is mandatory when designing QA systems, and usually goes in-line with a question classification or an identification of the question category. In some QA systems such as in (Moldovan and Harabagiu, 2000) the identification of the answer type is followed by the identification of the question focus.
Identification of the question category is usually based on the static rules which categorise the questions based on their syntax. For example, questions starting with *Where* would be in the different category from questions starting with *What*. This approach is used in various guises in many similar NLIs to ontologies such as ORAKEL (Cimiano et al., 2007), PANTO (Wang et al., 2007), Querix (Kaufmann et al., 2006), and AquaLog (Lopez et al., 2007). Our approach is different in that we try to avoid strict adherence to syntax, while engaging the user into the dialog in order to map certain syntactic structures into the ontology concepts. In addition, apart from ORAKEL, which can be customised through the user interaction, other mentioned systems build their lexicon dynamically from the ontology, and the downside of this approach is that the quality of the ontology strongly affects the system performance. With our user-system interaction approach we try to map the vocabulary used by the user to the one available in the ontology. The engagement of the user might be seen as an overload, however, making NLIs to ontologies or closed-domain QA systems with a reasonable performance usually requires customisation. If this customisation happens through the user-interaction process then this is less time-consuming than doing it manually by browsing the ontology and mapping relevant ontology concepts with human-understandable words.
While learning to map the syntax tree to the semantic meaning has not been extensively researched in the domain of NLIs to ontologies, several promising approaches have been tried and evaluated in some other domains such as for example NLIs to databases (Ge and Mooney, 2009). Supervised approaches such as learning the semantic parser based on statistical machine translation (Wong and Mooney, 2007), statistical disambiguation model (Ge and Mooney, 2009), and hidden-variable approach for learning to interpret sentences in context (Zettlemoyer and Collins, 2009) could all be seen as complementary to our approach which is semi-supervised.

6. Conclusion

In this paper we described our approach for identification of the answer type, which is implemented in our Natural Language Interface to Ontologies, called FREyA. We first run the algorithm which combines syntactic parsing with several heuristic rules. The output of this algorithm is further combined with ontology-based annotations, in order to identify the answer type. If necessary, the user is engaged in the dialog in order to solve ambiguities and precisely identify the answer type. Our evaluation with 250 questions from the Mooney Geoquery dataset shows that the answer type is correctly identified for 98.18% of questions, including 37.36% which required one dialog with the user.

7. Acknowledgments

We would like to thank Abraham Bernstein and Esther Kaufmann from the University of Zurich, for sharing with us Mooney dataset in OWL format, and J. Mooney from University of Texas for making this dataset publicly available.
This research has been partially supported by the EU-funded MUSING (FP6-027097) and LarKC (FP7-215535) projects.

8. References

Philipp Cimiano, Peter Haase, and Jörg Heizmann. 2007. Porting natural language interfaces between domains: an experimental user study with the orakel system. In *IUI '07: Proceedings of the 12th international conference on Intelligent user interfaces*, pages 180–189, New York, NY, USA. ACM.

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

H. Cunningham. 2005. Information Extraction, Automatic. *Encyclopedia of Language and Linguistics, 2nd Edition*, pages 665–677.

D. Damljanovic and K. Bontcheva. 2009. Towards enhanced usability of natural language interfaces to knowledge bases. In V. Devedzic and D. Gasevic, editors, *Special issue on Semantic Web and Web 2.0*, volume 6, pages 105–133. Springer-Verlag, Berlin, Germany.

Danica Damljanovic, Valentin Tablan, and Kalina Bontcheva. 2008. A text-based query interface to owl ontologies. In *6th Language Resources and Evaluation Conference (LREC)*, Marrakech, Morocco, May. ELRA.

Danica Damljanovic, Milan Agatonovic, and Hamish Cunningham. 2010. Natural Language Interfaces to Ontologies: Combining Syntactic Analysis and Ontology-based Lookup through the User Interaction. In *Proceedings of the 7th Extended Semantic Web Conference (ESWC 2010)*, Lecture Notes in Computer Science, Crete, Greece, June. Springer-Verlag New York Inc.

Christiane Fellbaum, editor. 1998. *WordNet - An Electronic Lexical Database*. MIT Press.

O. Ferret, B. Grau, M. Hurault-plantet, G. Illouz, L. Monceaux, I. Robba, and A. Vilnat. 2001. Finding an answer based on the recognition of the question focus.

Ruifang Ge and Raymond J. Mooney. 2009. Learning a compositional semantic parser using an existing syntactic parser. In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 611–619, Morristown, NJ, USA. Association for Computational Linguistics.

Catalina Hallett, Donia Scott, and Richard Power. 2007. Composing questions through conceptual authoring. *Computational Linguistics*, 33(1):105–133.

Esther Kaufmann and Abraham Bernstein. 2007. How useful are natural language interfaces to the semantic web for casual end-users? In *Proceedings of the Forth European Semantic Web Conference (ESWC 2007)*, Innsbruck, Austria, June.

Esther Kaufmann, Abraham Bernstein, and Renato Zumstein. 2006. Querix: A natural language interface to query ontologies based on clarification dialogs. In *5th International Semantic Web Conference (ISWC 2006)*, pages 980–981. Springer, November.

Dan Klein and Christopher D. Manning. 2002. Fast exact inference with a factored model for natural language parsing. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15 - Neural Information Processing Systems, NIPS 2002*, pages 3–10. MIT Press.

Vanessa Lopez, Victoria Uren, Enrico Motta, and Michele Pasin. 2007. Aqualog: An ontology-driven question answering system for organizational semantic intranets. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):72–105, June.

Dan Moldovan and Sanda M. Harabagiu. 2000. The structure and performance of an open-domain question answering system. In *Proceedings of the 38th Meeting of the Association for Computational Linguistics (ACL-2000)*, pages 563–570.

Raymond J. Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *In Proceedings of the 12th European Conference on Machine Learning*, pages 466–477.

Yun Niu, Graeme Hirst, Gregory McArthur, and Patricia Rodriguez-Gianolli. 2003. Answering clinical questions with role identification. In *Proceedings of the ACL 2003 Workshop on Natural Language Processing in Biomedicine*, pages 73–80.

Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. 2003. Towards a theory of natural language interfaces to databases. In *IUI '03: Proceedings of the 8th international conference on Intelligent user interfaces*, pages 149—157, New York, NY, USA. ACM.

Craig W. Thompson, Paul Pazandak, and Harry R. Tennant. 2005. Talk to your semantic web. *IEEE Internet Computing*, 9(6):75–78.

Chong Wang, Miao Xiong, Qi Zhou, and Yong Yu. 2007. Panto: A portable natural language interface to ontologies. In *The Semantic Web: Research and Applications*, pages 473—487. Springer.

Yuk Wah Wong and Raymond J. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-2007)*, pages 960–967. Association for Computational Linguistics.

Luke S. Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 976–984, Morristown, NJ, USA. Association for Computational Linguistics.