Web Semantics: Science, Services and Agents on the World Wide Web 19 (2013) 1–21

Contents lists available at SciVerse ScienceDirect

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

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

# Improving habitability of natural language interfaces for querying ontologies with feedback and clarification dialogues

Danica Damljanović [a,*], Milan Agatonović [b], Hamish Cunningham [a], Kalina Bontcheva [a]

[a] University of Sheffield, Department of Computer Science, Sheffield, United Kingdom

[b] Fizzback, London, United Kingdom

ARTICLE INFO

*Article history:*
Received 25 January 2011
Received in revised form
8 February 2013
Accepted 20 February 2013
Available online 5 March 2013

*Keywords:*
Natural language interfaces
Ontologies
Learning
Clarification dialogues
User interaction
Feedback

ABSTRACT

Natural Language Interfaces (NLIs) are a viable, human-readable alternative to complex, formal query languages like SPARQL, which are typically used for accessing semantically structured data (e.g. RDF and OWL repositories). However, in order to cope with natural language ambiguities, NLIs typically support a more restricted language. A major challenge when designing such restricted languages is habitability – how easily, naturally and effectively users can use the language to express themselves within the constraints imposed by the system. In this paper, we investigate two methods for improving the habitability of a Natural Language Interface: feedback and clarification dialogues. We model feedback by showing the user how the system interprets the query, thus suggesting repair through query reformulation. Next, we investigate how clarification dialogues can be used to control the query interpretations generated by the system. To reduce the cognitive overhead, clarification dialogues are coupled with a learning mechanism. Both methods are shown to have a positive effect on the overall performance and habitability.



## 1. Introduction

Recent years have seen a tremendous increase in structured data on the Web, with public sectors such as UK and USA governments opening their data to the public,[1] and encouraging others to build useful applications on top. At the same time, the Linked Open Data (LOD) project[2] continues to promote the authoring, publication and interlinking of new RDF graphs with those already in the LOD cloud [1]. In March 2009, around 4 billion RDF statements were available while in September 2010 this number increased to 25 billion, and continues to grow. This massive amount of data requires effective exploitation, which is now a great challenge largely due to the complexity and syntactic unfamiliarity of the underlying triple models and the query languages built on top of them. Natural Language Interfaces (NLIs) to rich, structured data, such as RDF and OWL repositories, are a viable, human-readable alternative.

The main challenges related to building NLIs are centred around solving the *Natural Language understanding* problem, *the data* that is being queried, and *the user*, and the way in which the user's information need is verbalised into a question.

Solving the *Natural Language understanding* problem includes grammar analysis, and solving language ambiguity and expressiveness, e.g. [2]. *Ambiguity* can be avoided through the use of a Controlled Natural Language (CNL): a subset of Natural Language (NL) that includes a limited vocabulary and grammar rules that must be followed. *Expressiveness* can be improved by extending the system vocabulary with the use of external resources such as WordNet [3] or FrameNet [4].

The second group of challenges is related to *the data* that is being queried, and building *portable* systems—those that can be easily ported from one domain or ontology to another without significant effort. According to [5], a major challenge when building NLIs is to provide the information the system needs to bridge the gap between the way *the user* thinks about the domain of discourse and the way *the domain knowledge is structured* for computer processing. This implies that in the context of NLIs to ontologies, it is very important to consider the ontology structure and content. Two ontologies describing identical domains (e.g., music) can use different modelling conventions. For example, while one ontology can use a datatype property `artistName` of class `Artist`, the other one might use instances of a special class to model the artist's name.[3]

* Corresponding author. Tel.: +44 1142221931.
*E-mail addresses:* danica.damljanovic@gmail.com, danica@kuatostudios.com (D. Damljanović), milan.agatonovic@gmail.com (M. Agatonović), h.cunningham@dcs.shef.ac.uk (H. Cunningham), k.bontcheva@dcs.shef.ac.uk (K. Bontcheva).

1 http://data.gov.uk and http://www.data.gov.

2 http://linkeddata.org.

3 See for example how the class `Alias` is used in the Proton System Module ontology: http://proton.semanticweb.org/.

1570-8268/$ – see front matter 
http://dx.doi.org/10.1016/j.websem.2013.02.002

Ontologies can be constructed to include sufficient lexical information to support a domain-independent query analysis engine. However, due to different processes used to generate ontologies, the extracted *domain lexicon* might be of varying quality. In addition, some words might have different meanings in two different domains. For example, "How big" might refer to `height`, but also to `length`, `area`, or `population`—depending on the question context, but also on the ontology structure. This kind of adjustments – or mappings from words or phrases to ontology concepts/relations – is performed during *customisation* of NLIs.

The third group of challenges is centred around *the users* and how they translate their information need into questions. While NLIs are intuitive, having only one text query box can pose difficulties for users, who need to express their information need through a natural language query effectively [6]. In order to address this problem, several *usability enhancement methods* have been developed with the aim to either assist users with query formulation, or to communicate the system's interpretation of the query to the user. In other words, the role of these methods is to increase the *habitability* of the system. Habitability refers to how easily, naturally and effectively users can use language to express themselves within the constraints imposed by the system. If users can express everything they need for their tasks, using the constrained system language, then such a language is considered habitable.

Our focus is on building *portable* systems that do not require a strict adherence to syntax—the supported language includes both grammatically correct and ill-formed questions, but also question fragments. We look at improving the *habitability* of such NLIs to ontologies through the application of *feedback* and *clarification dialogues*. We first discuss *habitability* and the four different domains that it covers in Section 2. We then describe how we model *feedback* relative to the specific habitability domains, and evaluate it in a user-centric, task-based evaluation (Section 3). Further on, in Section 4 we look at *clarification dialogues* and whether they can improve the specific *habitability* domains, by making the process of mapping a NL question into a formal query, transparent to the user. We combine the dialogue with a light *learning model* in order to reduce the user's cognitive overhead and improve the system's performance over time. We then examine the approach we have taken, which combines *clarification dialogues* with *learning*, in the controlled evaluation using the Mooney GeoQuery dataset.

## 2. Habitability

According to Epstein [7], a language is habitable if:

- Users are able to construct expressions of the language which they have not previously encountered, without significant conscious effort.
- Users are able to easily avoid constructing expressions that are not part of the language.

Another way of viewing habitability is as the mismatch between user expectations and the capabilities of an NLI system [8]. Ogden and Bernick [9] describe habitability in the context of four domains [9]:

- *The conceptual domain* of the language supported by the system describes the area of its coverage, and defines the complete set of objects and the actions which are covered. In other words, the conceptual domain determines *what* can be expressed by the system. Consequently, this domain is satisfied if the user does not ask about concepts which cannot be processed by the system. To cite the example from [9], the user could not ask "What is the salary of John Smith's manager?" if there is no information about *managers* in the system. The conceptual domain of the language can be expanded to inform the user that there is no information about *managers* in the system.
- *The functional domain* determines *how* a query to the system can be expressed. Natural language allows different ways of expressing the same fact, especially taking into account the knowledge of the listener and the context. The functional domain is determined by the number of built-in functions or knowledge the system has available. If, for example, the answer to a question requires combining several knowledge sources, the system itself might not be able to answer it and would require the user to ask two questions instead of one. A habitable system provides the functions that the user expects. Note that this is different from rephrasing the question due to unsupported grammar constructions, which is related to the syntactic domain.
- *The syntactic domain* of a language is determined by the number of paraphrases of a single command that the system understands. For example, to cite again the example from [9], the system might not be able to understand the question "What is the salary of John Smith's manager?" but, could be able to process a rephrased one such as "What is the salary of the manager of John Smith?".
- *The lexical domain* is determined by the words available in the lexicon. For example, in order to improve the coverage, many systems extend their lexicon through the use of external sources for finding synonyms.

For an NLI to be considered habitable, it should cover all four domains. Habitability is an important aspect of a system to measure because it can affect the usability of NLIs. By identifying why systems fail to be habitable, we can identify the ways to improve them [10].

One way to increase habitability is to use *usability enhancement methods* such as *feedback* and *clarification dialogues*. We first look at how feedback can improve the user's experience with an NLI, thus having an effect on habitability (Section 3). Further on, we look at using *clarification dialogues* to improve the habitability domains and make the process of mapping an NL question onto the formal query transparent; this gives the users control as they can influence the full interpretation of the query (Section 4).

## 3. Feedback

Showing the user the system's interpretation of the query in a suitably understandable format is called feedback. *Feedback* increases the user's confidence and in the case of failures, helps the user understand which *habitability domain* is affected. Several early studies [11,12] show that after receiving feedback, users are becoming more familiar with the system's interpretations and the next step is usually that they try to imitate the system's feedback language. In other words, returning feedback to the user helps them understand how the system interprets queries, therefore motivating them to use similar formulations and create queries that are *understandable* to the system.

Showing *feedback* can be useful for communicating the message between the user and computer clearly. This is comparable to human–human communication, where participants usually try to establish that the message they are trying to communicate is properly understood. This process is called *grounding*—as the users try to *ground* what is being said [13]. As pointed out by Clark and Brennan [13], humans seek evidence of understanding, which can either be *positive* or *negative*. *Negative evidence* is the evidence that they have not been understood, or heard, and if they find any, they attempt to *repair* it. If they fail to find any *negative evidence*, the assumption is that the other human understood the message correctly. However, often people search