ExtrAns
Answer Extraction from Technical Documents by Minimal Logical Forms and Selective Highlighting

Rolf Schwitter, Diego Mollá, Michael Hess

Department of Computer Science
Computational Linguistics
University of Zurich
Winterthurerstrasse 190
CH-8057 Zurich

Phone: +41-1-635 43 08
Fax: +41-1-635 68 09
E-Mail: {schwitter, molla, hess}@ifi.unizh.ch

Abstract

Logic-based answer extraction techniques present a solution to retrieve and mark those exact passages in a document that directly answer a natural language query. In contrast to pure information retrieval techniques that treat content words as isolated terms, answer extraction techniques exploit syntactic information in a document to a certain degree and consider semantic relations between function words and content words. Minimal logical forms (MLF) - specially designed for this task - represent the semantic relations of the sentences and point to the textual information in the document. MLFs consist of existentially closed atomic formulas and use reification of objects, eventualities and properties as a building principle. On account of their simple design MLFs proved to be computationally tractable and incrementally extensible in our answer extraction system ExtrAns. Unresolved structural ambiguities are represented by alternative MLFs. The theorem prover of ExtrAns finds all proofs for an (ambiguous) query and considers the frequency of a part of a MLF used during the proof as an indicator for the retrieval relevance. The actual retrieval relevance is reflected by selective highlighting in the document. The more often a part of a MLF that points to a specific phrase of a sentence is used for the proof, the more intensively this phrase is marked by the colouring scheme.

Keywords

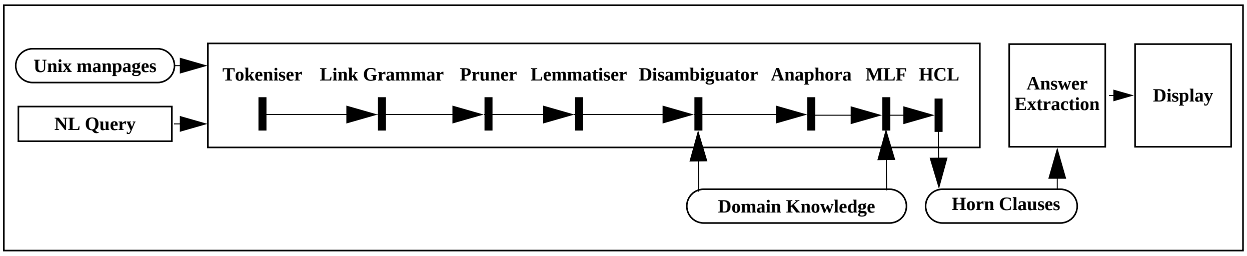
natural language processing, answer extraction, minimal logical form, disambiguation, selective highlighting

Answer Extraction Techniques

Finding information in documents still is a very cumbersome task. Standard information retrieval systems as well as the search engines on the Internet use techniques with two basic shortcomings. First, they treat both queries and documents as "bags of words", i.e. they ignore the information about the ways content words are syntactically combined, hence they do not distinguish for instance "design computer" and "computer design". Second, they return whole documents merely ranked in order of their statistical similarity to the query. If the user has highly specific information needs to solve a concrete problem in a technical domain this is unsatisfactory. The user will have to struggle through whole documents to find the relevant piece of information, and often he will not find it at all, since "design computer" and "computer design" simply do not denote the same thing. In recent years, alternative approaches which promise better results have been investigated [Burke et al. 97, Katz 97, Woods 97]. Among them is ***Answer Extraction***. This technique allows users to employ natural language to phrase their queries, and they return, as search result, those exact passages in the documents that contain the answer to the query. Possible applications are interfaces to machine-readable technical manuals or on-line help systems for complex software. The basic idea is to compute, by syntactic and semantic analysis, the core meaning of the documents and to represent it as minimal logical forms (MLF) whose individual terms contain pointers to those phrases and words in the documents whose meaning they represent. User queries are translated into theorems. A theorem prover tries to prove queries, by standard techniques (refutation), and from the pointers in those MLF terms that were used in a proof the phrases in the documents answering the queries can be determined. These phrases are highlighted through different colours and presented to the user in the context of the original documents. The user can thus spot the relevant passages of the documents at once and the context around the passages allows him to determine very quickly whether his question is actually answered by these phrases.

The ExtrAns System

In order to determine whether these techniques are really useful in a real world context we designed and implemented ExtrAns, an Answer Extraction system, that works over the Unix manual pages (*manpages*) [Mollá et al. 98]. The current version of ExtrAns runs over 500 unedited manpages. Here is an overview of ExtrAns:



The tokeniser of ExtrAns processes the manpages taking into account all formatting information about command names and named variables as well as domain-specific typographic conventions for path names and command options for further processing. The syntactic analysis for the documents is carried out by Link Grammar, a dependency-oriented grammar that consists of a large full form lexicon and a very fast parser [Sleator & Temperley 93]. A set of hand-crafted postprocessing rules prunes off some of the linkages that are clearly wrong. Since Link Grammar does not carry out any morphological analysis of the words, ExtrAns uses a lemmatiser that generates the lemma of the inflected words [Humphreys et al. 96]. Ambiguous attachments of prepositional phrases are disambiguated by a corpus-based approach trained with data from the manpages [Brill & Resnik 94]. Anaphoric references are resolved using an adaptation of a purely syntactic approach [Lappin & Leass 94]. To finally construct the MLFs, ExtrAns transforms the syntax structures generated by Link Grammar ("linkages") into a directed dependency structure. MLFs are converted into Horn clause logic (HCL): MLFs for data sentences into definite clauses by skolemising the variables and MLFs for user queries into negative clauses. The resulting knowledge base builds the starting point for the answer extraction. The quality of the proof for a user query is displayed by selective highlighting.

The lessons learned from this experiment were interesting in three respects. First, it became clear that the idea of locating individual phrases answering a question and presenting them in context is as powerful (and useful) as it is simple. Second, we found that a system with this kind of limited linguistic analysis can be built with the linguistic resources available today, and it can be scaled up to real world dimensions. Initially, we started with a subset of 30 Unix manpages, then extended the document basis to over 500, and found that no basic changes were needed. Third, and most interestingly, we found that it is possible to represent the relevant components of sentential meanings by a kind of minimal logical forms that can be extended, in incremental fashion, to become full-fledged logical forms if, and when, the need arises. In this paper we report on the first and the third point. [Mollá & Hess 99] inform about the scalability of ExtrAns.

Minimal Logical Forms

It turned out that, for answer extraction purposes in a narrow technical domain, most of the semantically difficult phenomena of natural language - from generalized quantifiers to the scope of negations to sentential connectors - do not have to be taken into account [Mollá et al. 98]. We thus use, as minimal logical forms, a representation of the propositional content of sentences, i.e. of those semantic relations that are expressed through function words and the basic case frames of content words. For the linguistic phenomena beyond those most basic ones that turn out to be relevant for the task at hand we add, incrementally, information in the form of additional first-order predicates. To this end we have to introduce a (potentially unlimited) number of reified concepts, following the ideas of [cf. Hobbs 85, Parsons 90, Stone 97]. The result is a flat set of existentially closed formulas, expressed by Horn clauses. By way of example, consider the following sentence (***cp*** is a Unix command and appears in the manpages in boldface):

> ***cp*** *copies files.*

The MLF of this sentence would be:

```
object(cp,x1), evt(copy,[x1,x2]), object(file,x2)
```

Here, `evt` stands of "eventuality" and x_i represent the obligatory objects involved in the copying event. Since even in the narrow domain considered, there occur sentences like

> ***cp*** *refuses to copy a file onto itself.*

in the manpages, we must be able to predicate over eventualities themselves. We thus need to represent this sentence as

```
holds(e1), object(cp,x1), evt(refuse,e1,[x1,e2]),
evt(copy,e2,[x1,x2]), object(file,x2), onto(e2,x2)
```

where we reify the eventualities of refusing and copying (`e1` and `e2`, respectively) and assert of the refusing event that it holds while the copying event must *not* be asserted to hold. Naturally, the sentence ***cp*** *copies files* above will now become

> **holds(e1)**, object(cp,**o1**,x1), evt(copy,**e1**,[x1,x2]), object(file,**o2**,x2)

For analogous reasons we also reify the concepts of objects being of a certain type and of properties to hold of certain objects. Thus, in order to distinguish intersective adjectives from intensional ones, we write

> ***cp*** *copies very long files.*
> holds(e1), object(cp,o1,x1), evt(copy,e1,[x1,x2]), object(file,o2,**x2**),
> prop(long,p1,**x2**), prop(very,p2,p1)

> ***cp*** *copies new files.*
> holds(e1), object(cp,o1,x1), evt(copy,e1,[x1,x2]), object(file,**o2**,x2),
> prop(new,p1,**o2**)

where the predicate modifier *very* requires us to predicate over the concept `p1` of `x2`'s being long, and where instances of objects (`x2`) as well as concepts (`o2`) are modified.

Prepositions are represented as homomorphic constants, i.e. the surface words are used as logical constants. The preposition *onto* thus gives `onto(e2,x2)`, if the prepositional phrase modifies an eventuality as below.

> ***cp*** *refuses to copy a file onto itself.*
> holds(e1), object(cp,o1,x1), evt(refuse,e1,[x1,e2]),
> evt(copy,e2,[x1,x2]), object(file,o2,x2), onto(e2,x2)

One of the consequences of working with flat structures is that surface negations and conjunctions are translated as regular predicates over reified concepts: for instance negations like *not* as `not(e1)` and conjunctions like *if* as `if(e1,e2)`.

__cp__ does not copy a file onto itself.

```
not(e1), object(cp,o1,x1), evt(copy,e1,[x1,x2]), object(file,o2,x2), onto(e1,x2)
```

If the user types y then __cp__ copies the files.

```
if(e1,e2), object(user,o1,x1), evt(type,e1,[x1,x2]), object(y,o2,x2),
object(cp,o3,x3), evt(copy,e2,[x3,x4]), object(file,o4,x4)
```

We can also make use of the incremental extensibility of our notation in cases where certain ambiguities cannot be locally resolved, as in the following example with nominal coordination with a distributive/collective ambiguity:

The mode and owner of filename2 are preserved if it already existed.

```
x1<$x3, x2<$x3, object(mode,o1,x1), object(owner,o2,x2), of(x3,x4),
object(filename2,o3,x4), evt(preserve,e1,[a1,x3]), object(anonym_object,o4,a1),
if(e2,e1), object(it,o5,x4), evt(exist,e2,[x4]), prop(already,p1,e2)
```

We use a lattice structure to represent the plural reading for coordinated structures by introducing a part-of operator <$ [Landman 91] but leave underspecified whether the reading is distributive or collective. Whenever, during a further processing step, more information becomes available these constraints can be added incrementally. Not only plural phenomena can be treated by this approach but also nominal compounds and even quantification including the difficult monotone decreasing quantifiers such as *few* or *no* [Hobbs 96].

Selective Highlighting as Retrieval Relevance

There are, however, cases where we cannot represent plural phenomena through underspecification as discussed above. In such cases we assert all readings. Those parts of the MLFs which are common to *all* readings will, naturally, be asserted more often than those that are not. Since we must always try to find *all* proofs to answer a given query we will use such assertions several times during the proof, but not all parts of them an equal number of times. The more often a part of an assertion is used the more relevant this information is taken to be for the associated phrase in the sentence. Consider, for instance, the *NAME*-entry in a manpage

rm, rmdir - remove files or directories.

which leads to the assertion of two predicates for the verb *remove* because of the comma-delimited enumeration in the subject position:

```
..., evt(remove,e1/P,[x3/P,x6/P])/P~[1,4,5,6,7], ...
..., evt(remove,e1/P,[x3/P,x6/P])/P~[2,4,5,6,7], ...
```

The attached lists (after the ~) indicates which words in the sentence led to the current interpretation: In the first line we make reference to *rm*, and in the second to *rmdir*. The proof of the query

How do I remove a directory?

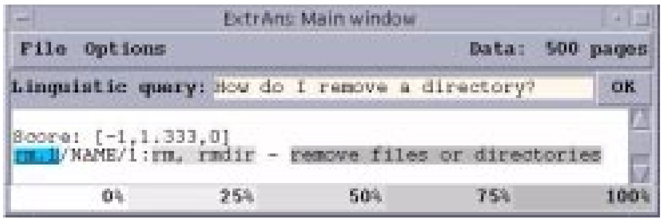
will thus extract once the underlined information

```
'rm.1'/'Name'/1/1
rm, rmdir - remove files or directories
```

and once

```
'rm.1'/'Name'/1/2
rm, rmdir - remove files or directories
```

In the first case *rm* was used during the proof, and in the second, *rmdir*. As a result, each of these terms gets a retrieval relevance value of 50% while the rest of the phrase gets 100%. ExtrAns converts this result into a graded colouring scheme and presents the result to the user by selective highlighting the relevant parts. If the user clicks on *rm.1/Name/1* in the graphical interface of ExtrAns (presented by a greyscale screen shot below) then the same result is displayed in the context of the document.



Linguistic query: How do I remove a directory?

```
Score: [-1,1.333,0]
rm.1/NAME/1:rm, rmdir - remove files or directories
```

This technique of selective highlighting was generalized for the treatment of unresolved structural ambiguities in ExtrAns. Similar to the plural example above all readings for an ambiguous sentence are asserted (which is in this case, logically speaking, wrong) and evaluated. That way, users are not confronted with several competing readings for an ambiguous sentence but with one sentence where ambiguities are reflected by different colours. This makes ambiguities far less obtrusive than all other ways to treat unresolvable ambiguities known to us.

If a user is not satisfied with the extracted answer, the search space can be made wider, in a stepwise manner: In a first step, synonyms are added to the query. In a second step, hyponyms are used. In a third step, the logical dependencies between all MLF terms are broken. And if all of these steps are unsuccessful a simple keyword search is performed.

Conclusions

Incrementally extensible MLFs seem to be a promising approach to keep the balance between computational tractability and expressivity. Presenting techniques like selective highlighting in context lead the users of the ExtrAns system directly to the information that satisfies their information need while unresolvable ambiguities are kept unobtrusively in the background.

References

[Brill & Resnik 94] E. Brill, P. Resnik, A rule-based approach to prepositional phrase attachment disambiguation, Proceedings of the 15th International Conference on Computational Linguistics (COLING'94), Vol. 2, pp. 998-1204, Kyoto, Japan, 1994.

[Burke et al. 97] R. D. Burke, K. J. Hammond, V. A. Kulyukin, S. L. Lytinen, N. Tomuro, S. Schoenberg, Question Answering from Frequently-Asked Question Files: Experiences with the FAQ Finder System, Technical Report TR-97-05, University of Chicago, Chicago, Illinois, June 1997.

[Hobbs 85] J. R. Hobbs, Ontological promiscuity, Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics, pp. 61-69, University of Chicago, 1995.

[Hobbs 96] J. R. Hobbs, Monotone decreasing quantifiers in a scope-free logical form. In K. van Deemter and S. Peters (eds.), Semantic Ambiguity and Underspecification, pp. 55-76, CSLI publications, Stanford, CA, 1996.

[Humphreys et al. 96] K. Humphreys, R. Gaizauskas, H. Cunningham, S. Azzam, GATE: VIE technical specifications, Technical Report, ILASH, University of Sheffield, 1996.

[Katz 97] B. Katz, From Sentence Processing to Information Access on the World Wide Web, AAAI Spring Symposium on Natural Language Processing for the World Wide Web, Stanford University, Stanford CA, 1997.

[Landman 91] F. Landman, Structures for Semantics, Kluwer, Dordrecht, 1991.

[Mollá et al. 98] D. Mollá, J. Berri, M. Hess, A Real World Implementation of Answer Extraction, Proceedings of the 9th International Conference and Workshop on Database and Expert Systems, Workshop "Natural Language and Information Systems" (NLIS'98), Vienna, 1998.

[Mollá & Hess 99] D. Mollá Aliod, M. Hess, On the Scalability of the Answer Extraction System "ExtrAns", to be presented at NLDB'99, 4th International Conference on Applications of Natural Language to Information Systems, June 17-19, University of Klagenfurt, Austria, 1999.

[Parsons 90] T. Parsons, Events in the Semantics of English: A Study in Subatomic Semantics, Current Studies in Linguistics, MIT Press, Cambridge, Mass., 1990.

[Sleator & Temperley 93] D. Sleator, D. Temperley, Parsing English with a Link Grammar, Third International Workshop on Parsing Technologies, August 1993.

[Stone 97] M. Stone, Ontological promiscuity revisited. Available over the Internet: <http://www.cis.upenn.edu/~matthew/cogsci.pdf.gz>, February 1997.

[Woods 97] W. A. Woods, Conceptual Indexing: A Better Way to Organize Knowledge, Technical Report TR-97-61, Sun Microsystems, Inc., Palo Alto, CA, 1997.