Towards Answer Extraction: An Application to Technical Domains

Fabio Rinaldi[*] and **James Dowdall**[*] and **Michael Hess**[*] and **Diego Mollá**[†] and **Rolf Schwitter**[†]

Abstract. The shortcomings of traditional Information Retrieval are most evident when users require exact information rather than relevant documents. This practical need is pushing the research community towards systems that can exactly pinpoint those parts of documents that contain the information requested. Answer Extraction (AE) systems aim to satisfy this need. This paper presents one such system (ExtrAns) which works by transforming documents and queries into a semantic representation called Minimal Logical Form (MLF) and derives the answers by logical proof from the documents. MLFs use underspecification to overcome the problems associated with a complete semantic representation and offer the possibility of monotonic, non-destructive extension.

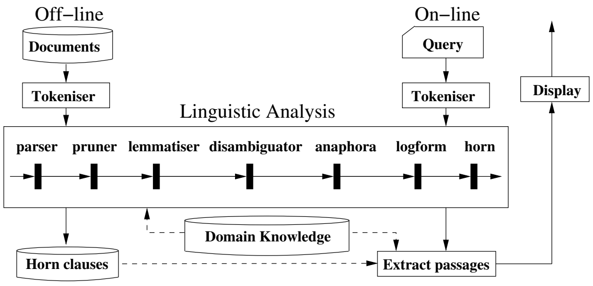

Figure 1. Architecture of the ExtrAns system

1 Introduction

The classical type of 'information need' solved by existing Information Retrieval (IR) applications has a number of shortcomings which new techniques such as Information Extraction and Answer Extraction aim at solving. Traditionally, it is assumed that IR systems have to find supporting documents on a particular topic and the problem of locating the relevant information within those documents is not dealt with. In fact, many authors have observed that traditional Information Retrieval should rather be called "Document Retrieval". The typical scenario application for IR techniques could be considered that of "Essay Writing", while the new approaches aim at a different scenario which could be called "Problem Solving".

Recently, some sections of the research community have focused their interest on systems which can not only locate relevant documents, but also pinpoint the exact piece of information that the user is interested in. During the past decade the Message Understanding Conferences have been a major arena for development in this field. The concept of Information Extraction has been gradually developed and refined so that today this is considered a separate and autonomous area of research. Typically such system can extract specific types of information predefined by the creators of the system. The simpler applications, like Named Entity extraction, have enjoyed considerable success. More complex applications, like template extraction and scenario extraction did not seem capable of improving significantly after reaching levels which were deemed interesting but not fully satisfactory. A fundamental problem with Information Extraction applications of the complex type (Template Extraction, Scenario Extraction) is that the system is normally tailored to the predefined templates and cannot easily adapt to different templates as would normally be required by a change of domain or the specific interests of the users (as defined by the templates).

Answer Extraction (also called Question Answering, or **QA**) is a recently developed field, which tries to solve some of the problems described above. Answer Extraction systems typically allow the user to ask arbitrary questions and aim at retrieving, in a given corpus, a small snippet of text which provides an answer to that questions. Research in this area has been promoted in the past couple of years by, in particular, the QA track of the TREC competitions [18, 21]. The participants in this competition have the opportunity to measure how well their systems can retrieve answers to a predefined set of questions from a very large collection of documents. They run their system on the given questions and return for each a ranked list of five answers in the form of pairs [document identifier, answer string]. The returned data are then evaluated by human assessors, who for each string have to decide whether it contains an answer to the question and whether the given document supports that answer.

One of the limitations of such evaluations has been that questions about rule-like or *definitional* knowledge (i.e. generic, intensional questions), such as "How do you stop a Diesel engine?" or "How does a Diesel engine work?", "When/where/how do typhoons form?" or "What is a typhoon?" have not received much attention so far.[1] In fact, it is precisely this type of question that users would direct at technical documents.

Besides there has been a strong focus on very large volumes of text, as typically seen in IR applications. In our own research we prefer to concentrate on "low volume/high value" data, with a gradual increase in volumes to follow later.

In this paper we present an Answer Extraction (AE) system (section 2) and its application to two different domains. After describing

[*] University of Zurich, Institute of Computational Linguistics, Winterthurerstrasse 190, CH-8057 Zurich, Switzerland . Email: {rinaldi, hess, dowdall}@ifi.unizh.ch

[†] Department of Computing, Macquarie University, Sydney, Australia. Email: {diego, rolfs}@ics.mq.edu.au

[1] While only a small number of them were included in QA track of TREC 8 and 9, in the most recent TREC 10 a significant number has been included.

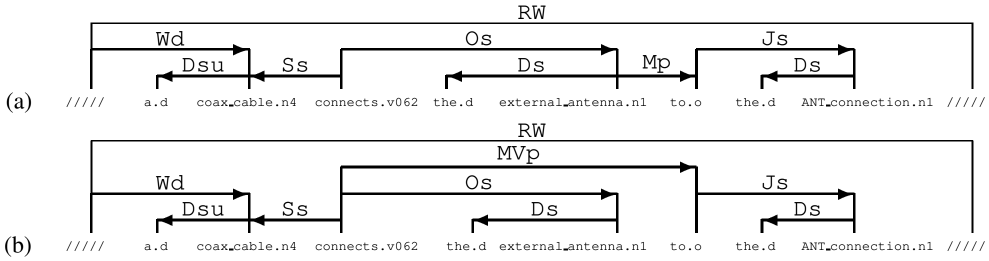

Figure 2. An example of LG output

in detail the syntactic (section 3) and semantic (section 4) processing components of the system, we will show how Answer Extraction is performed (section 5) and describe a comparison with a baseline IR system (section 6). Finally, we discuss the results and briefly survey related work (section 7).

2 The Architecture of ExtrAns

ExtrAns is a complex system which comprises several different modules (see figure 1), written in different programming languages. A given document collection is processed in an off-line stage, but the query is processed on-line. The same linguistic analysis is applied in both stages, transforming everything into a semantic representation called Minimal Logical Form (MLF).

The basic architecture has been tested over two contrasting technical domains. Originally, answers to arbitrary user questions were extracted from the Unix documentation files ("man pages"). The system covers a set of more than 500 unedited man pages, and answers questions such as *"which command can duplicate files ?"*. The flexibility of the MLFs allows the extraction of relevant answers (*"cp does not copy a file onto itself"*), not only strictly logical answers (*"cp copies files"*). The system can be tested on the project web page. [2]

More recently, the system has been used over the Airplane Maintenance Manuals (AMM) of the Airbus A320. The highly technical nature of this domain as well as an SGML-based format and a much larger size (120MB) than the Unix documentation, provided an important test-bed for the scalability and domain independence of the system.

Compared to the QA track of TREC these two domains represent small to medium sized document collections. An obvious advantage is the opportunity to process the entire document collection, rather than just selected paragraphs, in an off-line stage. As the data sets continue to grow in size this approach will quickly become too computationally expensive and paragraph indexing methodologies will need to be used. Currently, there is a paragraph selection procedure based on a loose matching between query concepts and the stored semantic representation of the document.

User queries are processed on-line and converted into MLFs (possibly expanded by synonyms) and proved by refutation over the document knowledge base. Pointers to the original text attached to the retrieved logical forms allow the system to identify and highlight those words in the retrieved sentence that contribute most to that particular answer [14]. An example of the output of ExtrAns can be seen in figure 3. When the user clicks on one of the answers provided,

[2] http://www.ifi.unizh.ch/cl/extrans/

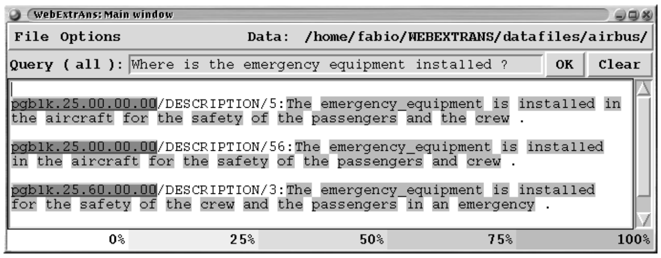

Figure 3. An example of the output of ExtrAns - query window

the corresponding document is displayed with the relevant passages highlighted.

When no direct proof for the user query is found (strict mode), the system is capable of relaxing the proof criteria in a stepwise manner. First, hyponyms will be added to the query terms, thus making the query more general but still logically correct. If that fails, the system will attempt approximate matching, in which the sentence with the highest overlap of predicates with the query is retrieved. The (partially) matching sentences are scored and the best fits are returned. In the case that even this method does not find sufficient answers the system will attempt keyword matching, in which syntactic criteria are abandoned and only information about word classes is used. This last step corresponds approximately to a traditional passage-retrieval methodology with consideration of the POS tags. It is important to note that, in the strict mode, the system finds only logically correct proofs (within the limits of what MLFs can represent; see below), i.e. it is a "high precision" AE system.

3 Syntactic Processing

The syntactic analysis uses the robust dependency-based parser Link Grammar (LG) [16], which is able to handle a wide range of syntactic structures [17]. Syntactically unresolvable ambiguities, such as prepositional phrase attachment or gerund and infinitive constructions, are treated with a corpus based approach [2]. Sentence-internal pronouns are dealt with using the anaphora resolution algorithm [11].

LG uses linkages to describe the syntactic structure of a sentence (see figure 2). Links connect pairs of words in such a way that the requirements of each word described in the sentences are satisfied, that the links do not cross, and that the words form a connected graph.

Despite some extensions at the lexical and syntactic level, processing the frequent occurrences of multi-word, domain specific terminology proved problematic for LG. The addition of a new module, capable of identifying these previously detected terms, ensures they are parsed as single syntactic units. This reduces the complexity of parsing the AMM, by as much as 50%. Also, the output of LG has been extended to include the direction of the linkages as this information is vital for anaphora resolution and semantic analysis.

As LG returns all possible parses, it is necessary to disambiguate among them [13]. The two possibilities for the prepositional phrase attachment returned in figure 2, will be reduced to (b) by the disambiguator as this linkage correctly identifies the dependency relations. The link `Wd` connects the subject *coax_cable* to the wall. The wall functions as a dummy word at the beginning of every sentence and has linking requirements like any other word. `Ss` links the transitive verb *connects* with the subject on the left, the verbal head on the right. The transitive verb and its direct object *external_antenna*, that acts as the head of a noun phrase, are connected by the `Os` link. `MVp` connects the verb to the modifying prepositional phrase. Finally, the link `Js` connects the preposition *to* with its object *ANT_connection*.

These dependency relations are used to generate the semantic representation of the sentence. LG has a robust component, parsing complex or ungrammatical structures, so that ExtrAns may still produce MLFs, extended with special predicates that mark the unprocessed words as "keywords".

Sentences that contain nominalizations are dealt with using a small hand-crafted resource (lexicon of nominalizations) [3] which helps us to cope with the most important cases, e.g. "to edit <a text>" ↔ "editor of <a text>"/"<text> editor". The system also includes hyponomy and synonymy relations based on the WordNet model.

4 Semantic Analysis

The Minimal Logical Forms (MLFs) of the documents and queries are the fundamental expression of their meaning within ExtrAns. The generation of MLFs is robust enough to treat very complex (even ungrammatical) sentences [14], and facilitates the semantic comparison of queries against documents. MLFs represent a powerful combination of selected reification and underspecification.

An important facet of the MLFs results from the flat expressions produced through reification, as proposed for instance in [9] or [5]. Where Hobb's ontologically promiscuous semantics reifies each predicate, MLFs restrict reification to only certain predicates: Objects, eventualities (events or states) and properties. In this way event modifiers, negations, higher order verbs, conditionals and higher order predicates can be represented.

MLFs use the main syntactic dependencies between words to express verb-argument relations, as well as modifier and adjunct relations. Extensive underspecification excludes complex quantification, tense and aspect, temporal relations, plurality and modality. One of the effects of this kind of underspecification is that several natural language queries, although slightly different in meaning, produce the same logical form.

The MLFs are expressed as conjunctions of predicates with all the variables existentially bound with wide scope. For example, the MLF of the sentence *"A coax cable connects the external antenna to the ANT connection"* is:

(1) holds($\boxed{1}$),
object(coax_cable, o2, [v3]),
object(external_antenna, o3, [v4]),
object(ANT_connection, o4, [v5]),
evt(connect, $\boxed{1}$, [v3, v4]),
prop(to, p1, [$\boxed{1}$, v5]),

[3] Similar to NOMLEX [12].

ExtrAns identifies three multi-word terms, translated into (1) as the objects: v3, a coax_cable, v4 an external_antenna and v5 an ANT_connection. The entity $\boxed{1}$ represents the 'connect' event involving two arguments, the coax_cable and the external_antenna. This reified event, $\boxed{1}$, is used again in the final clause to assert the event happens '*to*' v5 (the ANT_connection).

This is the utility of reification: yielding the additional arguments o2, o3, o4 and $\boxed{1}$ as hooks for additional modifiers to be attached to the entities they denote. Reification can be used to monotonically increment the underspecified MLF (1), without embedding arguments (preserving a flat structure), or destructively rewriting the original MLF.

For example, the expression *"A coax cable* ***securely*** *connects the external antenna to the ANT connection"* changes nothing in the original MLF, but additionally asserts: prop(securely, p8, $\boxed{1}$), that the event $\boxed{1}$ is *secure*. (1) only exploits the reified **event** but other, more complex sentences will need to refer to reified **objects** (non-intersective adjectives) or reified **properties** (adjective modifying adverbs).

5 Answer Extraction

ExtrAns finds the answers to the questions by forming the MLFs of the questions and then running Prolog's default resolution mechanism to find those MLFs that can prove the question. The logical form of the question *"How is the external antenna connected ?"* is:

(2) holds(v1),
object(external_antenna,o2,[v5]),
evt(connect,v1,[v4,v5]),
object(anonymous_object,v3,[v4]).

The variables introduced in a question MLF are converted into Prolog variables. The resulting MLF can be run as a Prolog query that will succeed provided that there has been an assertion in the text that the *external antenna* is connected to or by *something*. This *something* is the anonymous object of the query. A sentence identifier and a pointer (indicating the tokens from which the predicate has been derived) are attached to each predicate of a MLF in the knowledge base. This information matches against additional variables attached to the predicates in the question (not shown in the example above) and is eventually used to highlight the answer in the context of the document (see figure 3).

The use of Prolog resolution will find the answers that can logically prove the question, but given that the MLFs are simplified logical forms converted into flat structures, ExtrAns will find sentences that, logically speaking, may not be exact answers but are still relevant to the user's question, such as:

(3) a *"The external antenna must not be directly connected to the control panel."*
b *"Do not connect the external antenna before it is grounded."*
c *"The external antenna is connected, with a coax cable, to the ANT connection on the ELT transmitter."*
d *"To connect the external antenna use a coax cable."*

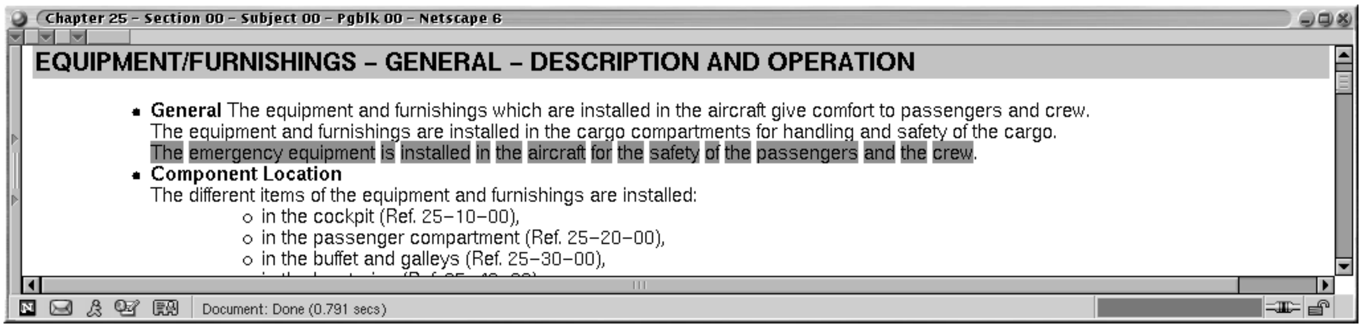

Figure 4. An example of the output of ExtrAns - document window

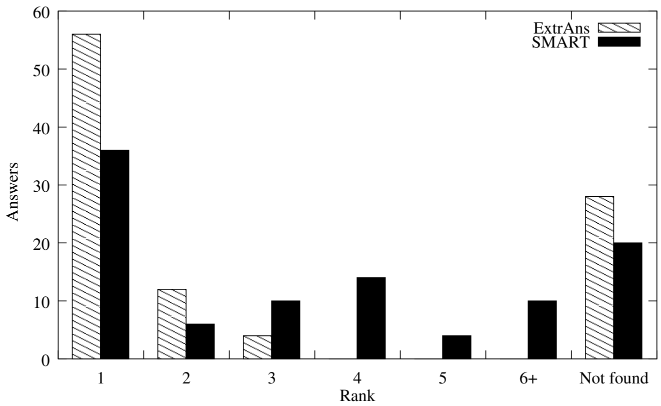

Figure 5. Answers at different ranks

The expressivity of the MLF is expanded through the use of meaning postulates of the type: *If x **is installed in** y, then x **is in** y*. This ensures that the query *"Where are the equipment and furnishings ?"*, extracts the answer *"The equipment and furnishings are installed in the cockpit"*.

In our view MLFs open up a potential path to a stepwise development of a question answering system by allowing monotonically incremental refinements of the representation without the need to destruct previous partial information. While MLFs specify the core meaning of sentences they leave underspecified those aspects of semantics that are less relevant or too hard to analyse, for the time being.

6 Evaluation

In order to set up an evaluation framework for our system, we decided to consider an IR system as a baseline, even if the standard measures of *precision* and *recall* are not ideal for an Answer Extraction system. In particular recall is significantly less important than precision, as the aim of such a system is to provide (at least) one correct answer, rather than all the possible answers in a given collection.

In the QA track of TREC a measure of precision that is commonly used is the Mean Reciprocal Rank (MRR). The Rank of a given result is the position in which the first correct answer is found in the output list of the system. Over a given set of answers MRR is computed as the mean of the reciprocals of the ranks for all the answers.

The particular evaluation that we present here[4] is targeted at the new application in the AMM domain. We devised 100 questions by selecting interesting passages from the manual and formulating questions of which those passages could be an answer. The questions were submitted to both ExtrAns and the selected IR system (SMART). While in general ExtrAns retrieves a short number of answers, that can be easily checked manually, SMART retrieves a ranked list of documents. As manual inspection of all the documents retrieved by SMART would be impossible, we decided to set an arbitrary threshold (at 10), i.e. if no valid answer was contained in the first ten retrieved documents, we classified it as "Not Found".

The diagram (figure 5) shows how many answers are found at each rank (1 to 5, answers from 6 to 10 are considered together). As it can be seen ExtrAns finds fewer answers than SMART (even ruling out all answers ranked > 10). Therefore recall would clearly be higher for SMART. However in the majority of cases, when ExtrAns does find the answer, it places it in the first position. Notice further that in some cases ExtrAns finds more than one valid answer for the same question (possibly in the same document).

There are very few cases where an answer at a lower rank is correct while answers at higher ranks for the same question are not. It does happen that ExtrAns retrieves incorrect answers together with the correct one, but in that case the correct one is almost always ranked first.

For the particular evaluation that we have presented our system would obtain a MRR of 0.63, which is a very good result if compared with results obtained in TREC. However we should stress that such a comparison is misleading, as our evaluation is far more restricted than those carried out in TREC. Besides, our system at the moment could not cope with very large volumes of data as seen in TREC.

In general, this evaluation leads us to conclude that ExtrAns can provide far higher precision than a generic IR system, at the price of a smaller recall. Recall alone however is not interesting. In the scenario that we consider it is important to locate quickly the precise answer. Relevant documents that are ranked poorly are likely to remain unnoticed by the user.

[4] We presented in [15] a different type of evaluation performed on the original application for the Unix man pages.

7 Discussion

IR techniques can be used to implement QA systems, by applying them at the passage or sentence level. Portions of text with the maximum overlap of question terms contain, with a certain probability, an answer. Standard preprocessing steps (removing stop words, "stemming" word forms, weighting keywords etc.) can be used to refine this basic method. However, systems that do not employ linguistic processing techniques and stick to the "bag of words" approach inherited from IR will never be able to distinguish different strings that contain the same words in different syntactic configurations and that therefore encode different meanings, such as "absence of evidence" and "evidence of absence".

Results from the two first TREC QA tracks [19, 21] showed clearly that traditional IR techniques are not sufficient for satisfactory Question Answering. When the answer is restricted to a very small window of text (50 bytes) systems that relied only on those techniques fared significantly worse than systems that employed some kind of language processing. More successful approaches employ special treatment for some terms (e.g. named entity recognition [7, 3]) or a taxonomy of questions [22, 1, 6, 10].

The standard methods used in IR to rank hits according to their relevance are no substitute for these techniques. Relevance in IR is almost invariably determined on the basis of the weights assigned to individual terms, and these weights are computed from term frequencies in the documents (or passages) and in the entire document collection (the *tf/idf*). Since this measure is blind to syntactic (and hence semantic) relationships it does not distinguish between hits that are logically correct and others.

It is interesting to observe how some of the systems that obtained good results in the QA track of TREC have gradually moved away from bag-of-words approaches and into NLP techniques, using semantic information. For instance, Falcon [8] (the best performing system in TREC 9) performs a complete analysis of a set of selected texts for each query and of the query itself and creates, after several intermediate steps, a logical representation inspired by the notation proposed by Hobbs (on which we also base our MLFs). The syntax analysis in Falcon is based on a statistical parser [4] while we use a dependency parser that computes all syntactically possible structures which we then filter according to a combination of hand-crafted rules and Brill and Resnik disambiguation procedure. A similarity between ExtrAns and Falcon is that both build a semantic form starting from a dependency-based representation of the questions.

As for the type of inferencing used, while ExtrAns uses standard deduction (proving questions over documents), Falcon uses an abductive backchaining mechanism, which can be used to provide a "logical proof" as a justification for the answer. Further it has an interesting module (which so far we do not have) capable of caching answers and detecting question similarity. In an environment where the same question (in different formulations) is likely to be repeated a number of times such a module can significantly improve the (perceived) performance of a QA system.

8 Conclusion

The QA track of TREC has proved that Natural Language Processing techniques cannot be dispensed of if relevant answers have to be pointed out precisely.

The meaning of both queries and documents must be taken into account, by syntactic and semantic analysis. Our fully functioning AE system, ExtrAns, shows that such applications are within the reach of present-day technology.

REFERENCES

[1] Eric Breck, John Burger, Lisa Ferro, Warren Greiff, Marc Light, Inderjeet Mani, and Jason Rennie, 'Another sys called Qanda', In Voorhees and Harman [21].

[2] Eric Brill and Philip Resnik, 'A rule-based approach to prepositional phrase attachment disambiguation', in *Proc. COLING '94*, volume 2, pp. 998–1004, Kyoto, Japan, (1994).

[3] C.L.A. Clarke, G.V. Cormack, D.I.E. Kisman, and T.R. Lynam, 'Question answering by passage selection (MultiText experiments for TREC-9)', In Voorhees and Harman [21].

[4] Michael Collins, 'A new statistical parser based on bigram lexical dependencies', in *Proceedings of the 34st Annual Meeting of the Association for Computational Linguistics, ACL-96*, pp. 184–191, (1996).

[5] Ann Copestake, Dan Flickinger, and Ivan A. Sag, 'Minimal recursion semantics: an introduction', Technical report, CSLI, Stanford University, Stanford, CA, (1997).

[6] David Elworthy, 'Question answering using a large NLP system', In Voorhees and Harman [21].

[7] Olivier Ferrett, Brigitte Grau, Martine Hurault-Plantet, and Gabriel Illouz, 'Qualc - the question-answering system of limsi-cnrs', In Voorhees and Harman [21].

[8] Sanda Harabagiu, Dan Moldovan, Marius Paşca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Gîrju, Vasile Rus, and Paul Morarescu, 'FALCON: Boosting knowledge for answer engines', In Voorhees and Harman [21].

[9] Jerry R. Hobbs, 'Ontological promiscuity', in *Proc. ACL'85*, pp. 61–69. University of Chicago, Association for Computational Linguistics, (1985).

[10] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-Yew Lin, 'Question answering in webclopedia', In Voorhees and Harman [21].

[11] Shalom Lappin and Herbert J. Leass, 'An algorithm for pronominal anaphora resolution', *Computational Linguistics*, **20**(4), 535–561, (1994).

[12] Adam Meyers, Catherine Macleod, Roman Yangarber, Ralph Grishman, Leslie Barrett, and Ruth Reeves, 'Using NOMLEX to produce nominalization patterns for information extraction', in *Proceedings: the Computational Treatment of Nominals, Montreal, Canada, (Coling-ACL98 workshop)*, (August 1998).

[13] Diego Mollá and Michael Hess, 'Dealing with ambiguities in an answer extraction system', in *Workshop on Representation and Treatment of Syntactic Ambiguity in Natural Language Processing*, pp. 21–24, Paris, (2000). ATALA.

[14] Diego Mollá, Rolf Schwitter, Michael Hess, and Rachel Fournier, 'ExtrAns, an answer extraction system', *T.A.L. special issue on Information Retrieval oriented Natural Language Processing*, (2000).

[15] Fabio Rinaldi, Michael Hess, Diego Mollá, Rolf Schwitter, James Dowdall, Gerold Schneider, and Rachel Fournier, 'Answer extraction in technical domains', in *Computational Linguistics and Intelligent Text Processing*, ed., A. Gelbukh, volume 2276 of *Lecture Notes in Computer Science*, 360–369, Springer-Verlag, (2002).

[16] Daniel D. Sleator and Davy Temperley, 'Parsing English with a link grammar', in *Proc. Third International Workshop on Parsing Technologies*, pp. 277–292, (1993).

[17] Richard F. E. Sutcliffe and Annette McElligott, 'Using the link parser of Sleator and Temperley to analyse a software manual corpus', in *Industrial Parsing of Software Manuals*, eds., Richard F. E. Sutcliffe, Heinz-Detlev Koch, and Annette McElligott, chapter 6, 89–102, Rodopi, Amsterdam, (1996).

[18] Ellen M. Voorhees, 'The TREC-8 Question Answering Track Evaluation', In Voorhees and Harman [20].

[19] Ellen M. Voorhees, 'The TREC-8 Question Answering Track Report', In Voorhees and Harman [20].

[20] Ellen M. Voorhees and Donna Harman, eds. *The Eighth Text REtrieval Conference (TREC-8)*. NIST, 2000.

[21] Ellen M. Voorhees and Donna Harman, eds. *Proceedings of the Ninth Text REtrieval Conference (TREC-9), Gaithersburg, Maryland, November 13-16, 2000*, 2001.

[22] W.A. Woods, Stephen Green, and Paul Martin, 'Halfway to question answering', In Voorhees and Harman [21].