Strategies for Web-based Cross-Language Question Answering [1]

Günter Neumann, Feiyu Xu, Bogdan Sacaleanu

LT–Lab, DFKI, Saarbrücken, Germany

Abstract

In this paper we present our current state-of-the art in the development of a hybrid QA architecture. In particular we present a strategy for open–domain web–based QA, and a strategy for open domain cross–language QA. In both cases the focus is on processing fact-based questions and exact answer strategies using the Web as primarily document source. Both strategies are realized using the same core technology, and have been implemented for German and English queries and documents, and tested for German Web pages.

Key words: web-based QA, exact answer, cross-language QA, QA architecture

1 Introduction

Our scientific view on the development of a generic question-answering (QA) system is that of a heterogeneous system architecture. The idea is that depending on the complexity of the query information (from simple fact-based questions, to relational template-based questions, to thematic-oriented questions) shallow or deep question answering (QA) strategies should be selected (or even mixed) which might involve different degrees of linguistic processing, domain reasoning or interactivity between a user and the system. In any of these cases, large-scale hybrid QA–technology are requested for handling

(1) open–domain as well as domain–specific information sources,

Email address: {`neumann|feiyu|bogdan`}`@dfki.de` (Günter Neumann, Feiyu Xu, Bogdan Sacaleanu).

[1] The work presented in this paper has been funded by the BMBF project Quetal, FKZ 01 IW C02. Many thanks to Olga Goldmann, Holger Neis, Ulrich Schäfer and Hubert Schlarb for their implementation support.

(2) cross-language queries and document pools,
(3) heterogenous information sources (large textual sources, WWW, pre–annotated corpus)

From a language technology point of view, important system design issues are:

(1) uniform linguistic core technologies for the modelling of a variable-depth “text-zooming”,
(2) through the integration of shallow and deep NLP components (motto: “shallow–first”, “deep on demand”), and
(3) data–driven parametrization and selection of system resources and information flow (based on large-scale Machine Learning).

In order to foster a bottom–up incremental system development, the initial focus should be on data directness, robustness, and scalability. In this paper we will report our current state–of–the art in the development of a generic hybrid QA–architecture. We will first present the blueprint of the envisaged QA–architecture. We will then describe two aspects in more detail:

- a strategy for open–domain web–based QA
- a strategy for cross–language QA

In both cases the focus is on processing fact-based questions and exact answer strategies using the Web as primarily document source. Both strategies have completely been implemented for German and English queries and documents, and tested mainly for German Web pages.

2 Overview of the Architecture

Two aspects of the architecture are uniform for query and document processing. These concerns the (shallow) linguistic analysis and the internal representation of QA objects.

Shallow NLP NL queries and documents are linguistically analyzed using ShProT, a shallow processing tool that consists of several integrated components: SPPC for tokenization and analysis of compound words (cf. [NP02]), TnT for part–of–speech tagging (cf. [Bra00]), Mmorph for morphological analysis (cf. [DG94]) and Chunkie for phrase recognition (cf. [SB98]). TnT and Chunkie are statistical based components which derive the linguistic entities, rules and generalizations from annotated corpora. The language models are based on the Penn treebank (for English) and the Negra treebank (for German). ShProT receives as input an ascii text and returns a stream of sentences each consisting of a sequence of tagged phrases and tagged wordforms. The tagged phrases actually define the type of the phrase (either NP or PP) and consists of a sequence of tagged wordforms. A tagged wordform contains the POS, and the lemma as determined by Mmorph. Named entity recognition (NER) is performed with two language specific components, SPPC for German (which consists basically of a set of manually specified finite state automata) and UnNER, a unsupervised NER learner for English based on [CS99].[2]

Bag of Objects Internally, queries and documents are uniformly represented as weighted sets of structured (possibly linked) objects in order to facilitate a robust and efficient comparison between queries and answer candidates. More formally, we call the set $B := \{O_1, \ldots, O_n; \alpha\}$ a *Bag–of–Objects* or short BoO consisting of n objects O_i and weight α. Each object O_i is a tuple of the form $\langle WF, Stem, PoS, NE, \alpha_i \rangle$, i.e., a structured object consisting of a word form, a lemma, part–of–speech, named entity and weight α_i.

The weight of a BoO is determined during the matching phase of the query with a candidate answer sentence or paragraph. The actual approach we are exploiting for comparing and merging two different BoOs is a variant of the *word overlap* method described in [LMRB01]. A word overlap (which is also a BoO in our case, hence we call it better an *object overlap*) is the subset of objects a query and an answer candidate have in common, i.e., the object overlap of two sentences s_1 and s_2 is $Ov_{s_1,s_2} := B_{s_1} \cap B_{s_2}$, where B_i is the BoO of s_i. The weight β of a object overlap Ov is determined as the sum over the weights α_i of the overlapping objects.[3] After Ov_{s_1,s_2} has been computed, the B_i obtain β as their weight, i.e., BoO with same object overlap have equal weight.

We also define the *overlap set* Os_q of a query q as the set of all BoOs of all candidate answer sentences which have the same object overlap with q, i.e., $Os_q := \{B_{s_1}, \ldots, B_{s_n}\}$, with: $Ov_{q,s_i} = Ov_{q,s_j}$ for $i \neq j$. This means that the overlap sets define equivalence classes over the set of possible answer candidates wrt. the set of objects each answer has in common with the query, i.e., query and sentences with same object overlap.

Note that we assume a sentence level representation, i.e., we assume that each document is internally represented as a stream of sentences, and that each sentence (including the query) is represented as a BoO. Thus, this representation is more closely related to those currently used in reading comprehension system (e.g., [HLBB99, Cha00, RT00]) than those used in information retrieval,

[2] Details are omitted because of lack of space.

[3] The weight of an individual object is currently specified a priori and is based on the word's part–of–speech.

cf. [BEYTW03]. Furthermore, it is easily possible to add to each element of the BoO a link to another element, e.g., in form of an index. In this way, for example, we can encode a dependency relation between (a subset) of the elements of a BoO or a flat predicate/argument structure. Since this helps us to uniformly represent a continuum of representations from more coarse-grained to more fine-grained structures, we are also able to apply the same overlap method on a higher level of abstraction or to use structurally more sensitive similarity measures between different BoOs.[4]

Query processing yields a query object which is represented as a BoO_q, which also contains the expected answer type *EAT*. The linguistic analysis of a query consists of two steps:

- shallow analysis using ShProT,
- clause level analysis of queries using a lexicalized tree grammar.

Query processing also involves query translation and expansion in which case we also make use of external services (e.g., EuroWordNet and online MT services; cf. sec. 4 for more details).

In our current system, we have specified manually a German and a English query grammar in form of a *Lexicalized Tree Grammar* (LTG). A query LTG consists of set of syntax/semantics oriented tree patterns which express mutual constraints for the identification of a question focus and an EAT. Here is an example of such an elementary tree:

```
 <tree id="6a" label="F-Wo" eat="LOCATION" freq="" prob="">
      <node label="PWAV">
          <node label="wo" type="TERM" anchor="YES"/>
      </node>
      <node label="VVFIN">
          <node label="schliessen" type="TERM" anchor="YES"/>
      </node>
      <node label="NE" nclass="PERSON" type="SUBST"/>
      <node label="NP" type="SUBST"/>
      <node label="PTKVZ">
          <node label="ab" type="TERM"/>
      </node>
</tree>
```

which would be applicable for a question like *Wo schloss Hillary Clinton das College ab?* (*Where did Hillary Clinton graduate college?*). A query grammar is applied on top of the shallow chunk analysis computed by first applying ShProT on the NL question. Note that nodes of type term are lexical anchors

[4] The underlying motivation here is similar to that described in [Mil99].

and nodes of type SUBST have to be expanded by substituting the node with a consistent (complete) phrase. Parsing of a query LTG is performed along the line of the method described in [Neu03].

The major motivation, why we have chosen a LTG approach is our future goal, to automatically extract a linguistically expressive but specific *query sub–grammar* form a large–scale general source grammar following the approach described in [Neu03], where we present a linguistically rich model of data–oriented parsing, called *HPSG–DOP*. The major idea behind HPSG–DOP is to automatically extract a *Stochastic* LTG from a Head-driven Phrase-Structure Grammar (HPSG) and a given corpus which can be processed much faster and robust then the original source grammar and which eases integration of domain knowledge more directly with syntactic constraints.

In some sense, the elementary trees of a LTG define clause–level patterns using lexical information about the question type and focus to constraint their applicability. Linguistically, an elementary tree of a LTG also describes a head–modifier relationship between the lexical anchors and the modifiers (basically the substitution nodes). Hence a derivation of a query analysis can also be used directly to uncover the dependency structure.

3 A strategy for Open–Domain Web–based QA

We briefly describe the major aspects of our strategy for open–domain web–based QA, a more detailed description can be found in [NX03]. For convenience, we will assume that the NL query processor has already computed the internal query object as outlined above.

All content words from BoO_q are passed to the Google search engine and the N-best documents (currently, N=50 is used) are further processed by ShProT. The main contributions of our approach for an open-domain Web-based answer extraction strategy are:

- An NE-directed voting technique by ranking all found NEs (independently from the fact whether they are relevant for the answer) using its term and document frequency (*multi-document* approach), with
- an answer extraction and ranking strategy using word/NE overlap between query expression and answer candidates as scoring function (cf. previous section).

The answer extraction process as a whole realizes a kind of *text zooming* method in the sense that we first identify the relevant paragraphs, then the relevant sentences, and then the relevant NE, i.e., the exact answer. In all of the subsequent steps NEs serve as *anchors* for the selection of the relevant textual window. In doing so, this strategy cannot only be used for extracting factoid answers, but can also be scaled-up for handling *list-based* and even *template-based* questions because of its inherent multi-document orientation, cf. sec. 6.

Finding the answer of a simple fact-based query basically means finding a single Named Entity (NE) – an instance of the expected answer type of the question, e.g., a person name for a who-question. We assume — in case of unary factoid questions — that a single sentence will contain the answer. However, since the Web pages returned by Google and further processed by SHPROT will contain many NE expressions in a single Web page as well as in *multiple* Web pages, simply iterating through all sentences to look for candidate answers is not appropriate.

In order to take advantage of the *redundancy* of NE expressions, we compute a weight for each recognized NE term as follows:

$r_{NE} = |DF_{NE}| \times (\alpha * TF_{NE}) + \sum_{i=1}^{|DF_{NE}|}(1 - \frac{r_i}{N})$

where, DF_{NE} is a list of documents containing the NE and ordered according to Google's ranking, TF_{NE} is the frequency of the NE, α is a smoothing factor, and r_i is the Google-rank of the ith document in DF. This means that an NE that occurs in more different documents will receive higher weight than an NE that occurs in fewer documents. Furthermore, NEs that occur in documents ranked higher by Google receive a larger weight than NEs occurring in lower ranked documents.

Once the weight for every recognized NE is computed, we construct an *inverted index* from the individual NEs to their positions in the original Web page. We further subdivide these indexed NEs by collecting all NEs of the same type into an individual list (e.g., a list of all found person names). These NE-lists are used for paragraph selection, which works as follows: for each NE from the NE-list which is type-compatible with the expected answer type of the current question (e.g., person) we determine each of its position P_{NE} in the original Web pages and extract an P_{NE}-centered window $S_1S_2S_{NE}S_3S_4$, where S_{NE} is the sentence containing the NE, and S_i are the adjacent sentences. For each triple $S_1S_2S_{NE}$, $S_2S_{NE}S_3$ and $S_{NE}S_3S_4$, a weight is computed based on the number of containing content words of the question and the distance of each identified content word from P_{NE}. The highest scored triple is selected as a candidate paragraph.

Sentence ranking Note that we consider each occurrence of an NE in the selected document set. Redundancy comes into play here, because it might be that different paragraphs from different documents which contain NE, differ only in view wordings because they contain the same or very similar sentences. We now describe, how we can collapse similar sentences occurring in multiple documents into a single equivalence class. These classes are then used as the basis for selecting and ranking answer candidates. The core idea is to collect all sentences that have the same rank into one group. We denote such a group as a sentence-based equivalence class. In principle, the rank is determined by computing the overlap of tokens and NEs between the query and each sentence. The scoring function *EC* used in our approach for building and ranking sentence-based equivalence classes is an extension of the one described in [LMRB01]. *EC* is defined as follows:

$$EC = (olToken + olNE) \times (1.5 + \frac{\sum_{i=1}^{n} r_{NE(EAT)_i}}{n})$$

Thus, we consider both the number of overlapped word forms or tokens (denoted as *olToken*), and the number of overlapped named entities (*olNE*). Note that this cannot include named entities that have the same type as the expected answer type (EAT) because the EAT actually serves as a typed variable in the question. However, we consider the weight of each NE that is compatible with the expected answer type (denoted as $r_{NE(EAT)}$). For n instances of the expected answer type occurring in a sentence (which means that the sentence has ambiguous answers), we use the average weight of the ambiguous NE's. In summary, a sentence has higher relevance than another one, if it shares more common words and named entities with the question, and if it contains instances of the expected answer type with high weights. The EAT compatible NEs are stored in a list and associated with the sentence equivalence class. These are chosen as exact answers ranked according to their weights. For example, for the question

Welches Pseudonym nahm Norma Jean Baker an?
Which pseudonym did Norma Jean Baker use?

the system returns a list of equivalence classes of sentences, and associated list of instances of ranked answer type named entities. Below, one equivalence class is shown (abbreviated as *eclass*) for the above question. It contains only one sentence candidate, where two ranked person names ("Marilyn Monroe" and "Norma Jean Mortenson") can be potential exact answers. The person name with highest weight is "Marilyn Monroe", which is selected as the best exact answer in this case.

```
<eclass rank='7.254032855348923'>
<sentence url='http://www.beatlesfan.de/alle.htm'>
   Marilyn Monroe war der Kuenstlername von Norma Jean Mortenson,
   auch bekannt als Norma Jean Baker.
</sentence>
```

```
 <exact-answer  type ='PERSON'>
  <name rank='0.6029282321968642'>
    Marilyn Monroe </name>
  <name rank='0.024088195477597402'>
    Norma Jean Mortenson </name>
 </exact-answer>
</eclass>
```

In other words, this means that a sentence-based equivalence class performs a further “zooming” step such that it collects from all documents those sentences together which have the highest degree of similarity wrt. the query where similarity is determined by the scoring function EC. The sentences of a class are then used to construct the ranked list of EAT compatible named entities NE(EAT). These then serve as the corresponding question’s answer candidates from which the k-best are chosen and presented to the user. In some other sense, the process of determining the ranked list of exact answers can also be interpreted as a specific kind of merging of those NE’s that are EAT-compatible and occur in similar sentences (of multiple documents). Thus our approach can also be used for handling list-based questions of the sort *Name 22 cities that have a subway system.*

Experiments The strategy has been tested using German Web pages. Our German question-answer pairs have been extracted from a popular quiz book. Currently, we have considered 39 questions of two answer types: person and location. For comparison we compare our results with the top five Google snippets, which we treat as answers extracted by Google. For example, Google obtained an average MMR=0.103 for all person questions, where our approach obtains MMR=0.212 (for exact answers), and MMR=0,216. A similar behavior was found for location names (MMR(Google)=0.092, MMR(our approach, exact answers)=0,135).

4 A Strategy for Cross-language QA

In this section we are going to describe in more detail how question translation and expansion is performed using the BoO approach as outlined above. The basic idea of our approach is to combine the results of external MT–services with the results found in WordNet on the level of our BoO approach. Currently, we translate the German language question to the English language of the document collection by means of machine translation techniques. The system accounts for the coverage issue by using three different translation services: FreeTranslation, Altavista and Logos. The results of translating the original German question are used in generation of BoO collections of English objects,

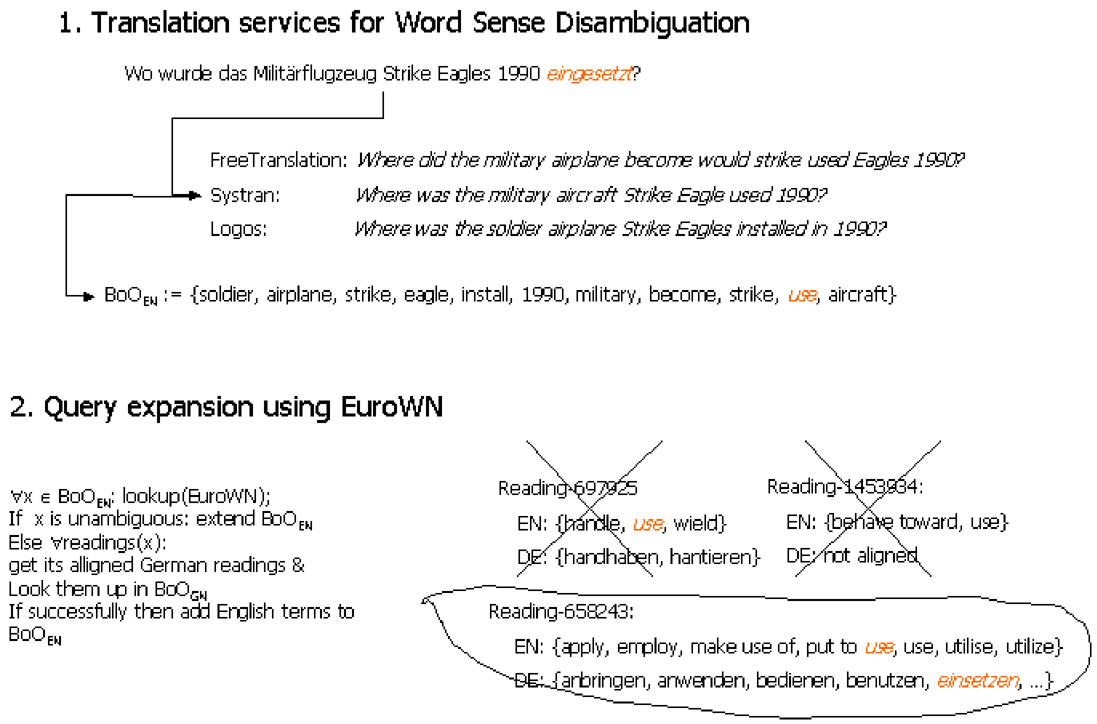
1. Translation services for Word Sense Disambiguation
Wo wurde das Militärflugzeug Strike Eagles 1990 eingesetzt?
FreeTranslation: Where did the military airplane become would strike used Eagles 1990?
Systran: Where was the military aircraft Strike Eagle used 1990?
Logos: Where was the soldier airplane Strike Eagles installed in 1990?
BoO_EN := {soldier, airplane, strike, eagle, install, 1990, military, become, strike, use, aircraft}
2. Query expansion using EuroWN
∀x ∈ BoO_EN: lookup(EuroWN);
If x is unambiguous: extend BoO_EN
Else ∀readings(x):
get its alligned German readings &
Look them up in BoO_GN
If successfully then add English terms to BoO_EN
Reading-697925
EN: {handle, use, wield}
DE: {handhaben, hantieren}
Reading-1453934:
EN: {behave toward, use}
DE: not aligned
Reading-658243:
EN: {apply, employ, make use of, put to use, use, utilise, utilize}
DE: {anbringen, anwenden, bedienen, benutzen, einsetzen, ...}

Fig. 1. The structure of cross-language QA strategy.

which are further on target of the query expansion module. Expansion is being achieved only through semantic variations using WordNet-like resources, whereby a pseudo word-sense-disambiguation task using the German original question and its English translations is being applied.

Figure 4 illustrates the functioning of the question translation and expansion module by means of the example question:

Wo wurde das Militärflugzeug Strike Eagles 1990 eingesetzt?

Given the translations, a BoO collection of open-class normalized words has been created which is denoted BoO_{EN} in the figure, part one (for convenience we abbreviate the object elements by means of their lexemes). For the expansion task we have used the German and English wordnets aligned within the EuroWordNet lexical resource. Our goal was to extend the English BoO collection with synonyms for the words that are present in the wordnet. Considering the ambiguity of words, a WSD module was required as part of the expansion task. For this purpose we have used both the original question and its translations, leveraging the reduction in ambiguity gained through translation. In figure 4 (part 2) the current query expansion method is presented and illustrates by examples. Following the question expansion task, the BoO collection has been enriched with new words that are synonyms of the un–ambiguous English words and by synonyms of those ambiguous words, whose meaning(s) have been found in the original German question.

Initial experiments using only one translation service unveiled the limitation imposed by the coverage problem: inadequate or no translations (e.g., some name of countries that were different in German and English). Extending the translation module with two further services, the results improved and pointed out the advantage of indirectly using it for question expansion as well, as different translations can generate synonym words. The approach has also been applied to a textual corpora, and we have participated at this years Clef. However, since this was our first time we participated in such a competition, the focus was on system implementation rather than system tuning, so the result of 15% coverage is enough positive motivation.

5 Related Work

There are only very view systems which process the language pair German/English. AnswerBus developed by [Zhe02] also allows German input queries. However the queries have to be translated (using the translator provided through Babelfish) into English because answer extraction is only performed for English Web pages. Furthermore, AnswerBus does not compute exact anwers, but complete sentences. In contrast, our system can also process German web pages, and is able to return exact answers. Our experiments have approved the experiences reported in [BLM^{+}01], [CCKL00], [KEW01], [LMRB01] and [Lin02], that the redundancy plays an important role for WebQA. The NE-ranking measure that we have developed is similar to that described in [CCKL00], which weights candidate answer terms by the integration of corpus frequency into the scoring function.

6 Future work

So far we have making use of shallow NLP components for query and document processing. For analysing the clause structure of WH–questions we are using specialized lexicalized tree grammars. In the next version of the system, we plan to make use of an integrated approach of shallow and deep NLP for question processing following our HPSG–DOP approach outlined above. This will help us to achieve a better accuracy for the recognition of fine-grained expected answer types. Furthermore, our BoO based approach allows us very easily to express multi-fact (or template-based) questions (by specifying more than one expected answer type of different type). This could also be viewed as specifying a kind of on demand template (who did what when where). In this case our paragraph selection and sentence ranking process would be extended to return partially filled templates that have to be merged in a later step to find candidate templates. Merging will be done under control of a domain–specific ontology in the sense, that it defines "ontological wellformedness". An initial approach has already been realized, however only for a very restricted scenario, cf. [KBB+01]. We hope to transform this approach into our hybrid QA system.

References

[BEYTW03] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 3:1183–1208, 2003.

[BLM+01] Eric Breck, Marc Light, Gideon S. Mann, Ellen Riloff, Brianne Brown, Pranav Anand, Mats Rooth, and Michael Thelen. Looking under the hood: Tools for diagnosing your question answering engine. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-2001) Workshop on Open-Domain Question Answering*, 2001.

[Bra00] Thorsten Brants. Tnt – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, Seattle, WA., 2000.

[CCKL00] Charles Clarke, Gordon Cormack, Derek Kisman, and Thomas Lynam. Question answering by passage selection (multitext experiments for TREC-9). In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, 2000.

[Cha00] E. Charniak et al. Reading comprehension programs in a statistical language processing class. In *Proceedings of the ANLP/NAACL 2000 Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, 2000.

[CS99] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.*, Association for Computational Linguistics, 1999.

[DG94] D. Petitpierre and G. Russell Mmorph - the multext morphology program. Technical report, ISSCO, University of Geneva, 1994.

[HLBB99] L. Hirschman, M. Light, E. Breck, and J. Burger. Deep read: A reading comprehension system. In *Proceedings of the 37th An-*

nual Meeting of the Association for Computational Linguistics., 1999.

[KBB$^+$01] M. Klettke, M. Bietz, I. Bruder, A. Heuer, D. Priebe, G. Neumann, M. Becker, J. Bedersdorfer, H. Uszkoreit, A. Maedche, S. Staab, and R. Studer. Getess - ontologien, objektrelationale datenbanken und textanalyse als bausteine einer semantischen suchmaschine. *Journal Datenbank-Spektrum*, 1(1):14–24, 2001.

[KEW01] Cody Kwok, Oren Etzioni, and Daniel S. Weld. Scaling question answering to the Web. In *Proceedings of the Tenth International World Wide Web Conference (WWW10)*, 2001.

[Lin02] Jimmy Lin. The Web as a resource for question answering: Perspectives and challenges. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-2002)*, 2002.

[LMRB01] Marc Light, Gideon S. Mann, Ellen Riloff, and Eric Breck. Analysis for elucidating current question answering technology. *Natural Language Engineering*, 7(4), 2001.

[Mil99] David Milward. Towards a robust semantics for dialogue using flat structures. In *Proceedings of Amsteogue '99. Workshop on Semantics and Pragmatics of Dialogue*, Amsterdam University, 1999.

[Neu03] Günter Neumann. Data-driven approaches to head-driven phrase structure grammar. In Rens Bod, Remko Scha, and Khalil Sima'an, editors, *DATA-ORIENTED PARSING*. CSLI Publications, University of Chicago Press, 2003.

[NP02] Günter Neumann and Jakub Piskorski. A Shallow text processing core engine. *Computational Intelligence*, 18(3):451–476, 2002.

[NX03] Günter Neumann and Feiyu Xu. Mining Answers in German Web Pages. In *Proceedings of The International Conference on Web Intelligence (WI 2003)*, Halifax, Canada, October 2003.

[RT00] E. Riloff and M. Thelen. A rule-based question answering system for reading comprehension tests. In *Proceedings of the ANLP/NAACL 2000 Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, 2000.

[SB98] Wojciech Skut and Thorsten Brants. A maximum entropy partial parser for unrestricted text. In *6th Workshop on Very Large Corpora*, Montreal, Canada, August 1998.

[Zhe02] Zhiping Zheng. AnswerBus question answering system. In *Proceeding of 2002 Human Language Technology Conference (HLT 2002)*, 2002.