The LogAnswer Project at ResPubliQA 2010

Ingo Glöckner[1] and Björn Pelzer[2]

[1] Intelligent Information and Communication Systems Group (IICS), University of Hagen, 59084 Hagen, Germany
ingo.gloeckner@fernuni-hagen.de

[2] Department of Computer Science, Artificial Intelligence Research Group University of Koblenz-Landau, Universitätsstr. 1, 56070 Koblenz
bpelzer@uni-koblenz.de

Abstract. The LogAnswer project investigates the potential of deep linguistic processing and logical reasoning for question answering. The paragraph selection task of ResPubliQA 2010 offered the opportunity to validate improvements of the LogAnswer QA system that reflect our experience from ResPubliQA 2009. Another objective was to demonstrate the benefit of QA technologies over a pure IR approach. Two runs were produced for ResPubliQA 2010: The first run corresponds to LogAnswer with standard configuration. The accuracy of 0.52 and c@1 score of 0.59 witness that LogAnswer has matured (in 2009, accuracy was 0.40 and c@1 was 0.44). In the second run, a special index that only indexes terms from the definiendum of definitions was used for answering definition questions. The resulting accuracy was 0.54 with c@1 score 0.62. For definition questions, accuracy increased by 21%. The deep linguistic analysis of LogAnswer and its validation techniques made a substantial difference compared to a pure IR approach. Using the retrieval stage of LogAnswer as the IR baseline, we found a 27% gain in accuracy and 37% gain in c@1 due to the powerful validation techniques of LogAnswer.

1 Introduction

The LogAnswer project investigates the potential of deep linguistic processing and logical reasoning for question answering.[3] The LogAnswer QA system developed in this project was first presented in CLEF 2008 [6], where it took part in the QA@CLEF track for German. After consolidation it participated in ResPubliQA 2009 where it was the third-best system under the *C@1 / Best IR baseline* metric. Still, the paragraph selection of LogAnswer were only slightly better than the very strong retrieval baseline. The paragraph selection (PS) task of ResPubliQA 2010 has offered the opportunity to validate improvements of the LogAnswer QA system that reflect our experience from ResPubliQA 2009:

[3] Funding of this work by the DFG (Deutsche Forschungsgemeinschaft) under contracts FU 263/12-2 and GL 682/1-2 (LogAnswer) is gratefully acknowledged. Thanks to Tim vor der Brück for his n-gram recognizer, and to Sven Hartrumpf for adapting the WOCADI parser.

– LogAnswer showed a very low accuracy for DEFINITION questions (16.8%). One reason for the low accuracy for definition questions was the restrictive way in which queries to the passage retrieval system were constructed; another problem was that the domain-specific way in which definitions are expressed in regulations was not recognized by LogAnswer. In ResPubliQA 2010, we have addressed these problems by improving the use of the retrieval system for definition questions and by building a dedicated definition index that also covers definitions in the form typically found in regulations.
– LogAnswer also showed a low accuracy for PROCEDURE questions (29.1%). We now tackle the difficulty to recognize sentences that describe a procedure by additional procedure triggers whose presence in the text marks a sentence as expressing a procedure. Moreover the question classification for PROCEDURE questions was incomplete (only 20.3% of these questions were recognized as such), so recognition rules had to be added to close this gap.
– LogAnswer depends on the parsing quality of the linguistic analysis stage since only sentences with a full parse allow a logic-based validation. The document collections of ResPubliQA with their administrative language are very difficult to parse, though – in the last year, the parser used by LogAnswer only managed to find a full parse for 26.2% of all sentences. Therefore one of our goals for ResPubliQA 2010 was to improve the parsing rate for JRC Acquis, and to ensure acceptable results for the new Europarl collection.
– In order to achieve a general performance improvement, we decided to try a more thorough use of compound decomposition, knowing that compound nouns abound in administrative texts written in German.

Apart from evaluating the current state of the LogAnswer prototype, our main objective was that of demonstrating a clear advantage of using QA technologies over a pure IR approach. We also wanted to show that LogAnswer can cope with the novel challenges of ResPubliQA 2010, i.e. with the Europarl corpus and with OPINION questions.

In the paper, we first explain how the LogAnswer QA system works and how we prepared it for ResPubliQA 2010. We then present the results of LogAnswer on the ResPubliQA question set for German. The discussion of these results and some additional experiments will highlight the strengths and some remaining weak points of the system. We conclude with a summary of the progress made.

2 System Description

LogAnswer rests on a deep linguistic analysis of the document collections by the WOCADI parser [8]. The pre-analysed sentences are stored in a Lucene index,using (synonym normalized) word senses from each sentence and occurrences of answer types (like PERSON) as index terms. Questions are also parsed by WOCADI. The (synonym normalized) word senses in a question and its expected answer type are used for retrieving linguistic analyses of 200 sentences. Various shallow features judging the question/snippet match are computed (e.g.

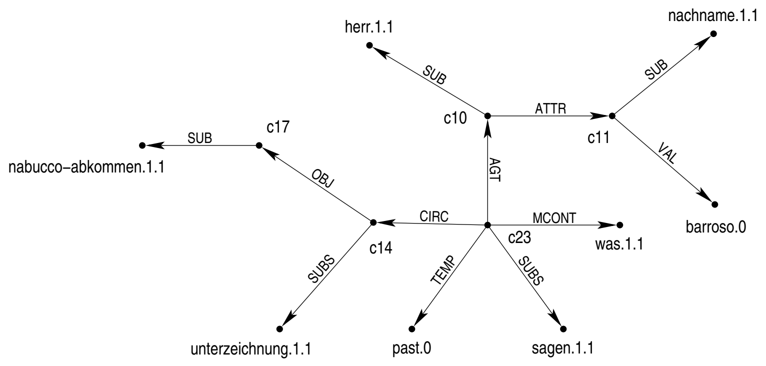

Fig. 1. MultiNet representation of the example question *Was hat Herr Barroso bei der Unterzeichnung des Nabucco-Abkommens gesagt?*

lexical overlap). For sentences with a full parse, a relaxation proof of the question from sentence and background knowledge provides additional logic-based features [3]. The best answer sentence is selected by a validation model based on rank-optimizing decision trees [5]. If the validation score of the best sentence exceeds a quality threshold, then the sentence is expanded into the final answer paragraph; otherwise 'no answer' is shown. The basic setup of the system is the same as described in [7]. In the following we detail some of the processing stages in the Q/A pipeline of LogAnswer, using a fixed question as a running example.

2.1 Linguistic Analysis of the Question

The questions are first subjected to a linguistic analysis, using the WOCADI parser for German [8]. WOCADI generates a meaning representation of the question in the semantic network formalism MultiNet [9]. Let us consider question #119 from the ResPubliQA 2010 test set for German:

> *Was hat Herr Barroso bei der Unterzeichnung des Nabucco-Abkommens gesagt?* (What did Mr Barroso say at the time of signing the Nabucco agreement?)

The relational structure[4] of the MultiNet representation generated for the example question is shown in Fig. 1. The indexed symbols like *sagen.1.1* are word sense identifiers. The MultiNet representation provides the basis for question classification and for the subsequent generation of a logical query.

Note that a complete MultiNet representation is only available if WOCADI finds a full question parse. This requirement was not problematic since WOCADI achieved a full parse rate of 97.5% on the ResPubliQA 2010 question set.

Apart from generating a semantic representation, the parser also provides the results of its morphological and lexical analysis stage. Note that the inclusion

[4] The labeling of nodes with additional 'layer attributes' is not shown for simplicity.

of name lexica allows the parser to tag named entities in the text by types such as *last-name* (the family name of person), etc. WOCADI further provides information on the decomposition of compound words that occur in a sentence. In our running example, it finds out that *nabucco-abkommen.1.1* is a regular compound built from *abkommen.1.1* (agreement) and *nabucco* (an unknown word that could represent a name *nabucco.0*, or a regular concept *nabucco.1.1*).

This kind of morpho-lexical and named-entity information is also available for sentences with a partial or failed parse and can thus be used for implementing fallback methods that replace a logical validation in the case of a parsing failure.

2.2 Question Classification

A rule-based question classification is used to determine the expected answer type of the question and to identify the descriptive core of the question. The expected answer types known to LogAnswer are a refinement of the question categories of ResPubliQA. Apart from OPINION, PROCEDURE, PURPOSE, REASON, and DEFINITION questions, the system distinguishes several types of factoid questions such as *city-name*, *mountain-name*, *first-name*, *last-name*, *island-name*, etc. These types correspond to the supported named entity types that WOCADI recognizes in the text.

For ResPubliQA 2010, the existing rule base for question classification was extended. In particular, the coverage of rules for PROCEDURE and PURPOSE questions was improved. In order to recognize compound triggers like *Arbeitsverfahren* (working procedure) or *Hauptaufgabe* (main task), LogAnswer now treats all nominal compounds that modify a known trigger word as additional trigger words for the corresponding question type. Finally, 25 rules for recognizing OPINION questions were added. The resulting rule base now comprises 240 classification rules, compared to 165 rules used for ResPubliQA 2009.

2.3 Retrieval of pre-analyzed sentences

Experiments with a paragraph-level and document-level segmentation of the texts have shown no benefit over sentence segmentation in the ResPubliQA 2009 task [7]. Therefore we decided to work with a simple sentence-level index.

Prior to indexing, all documents in the considered JRC Acquis and Europarl collections[5] must be analyzed by WOCADI. In order to achieve an acceptable parsing rate, some automatic regularizations of the documents were performed (such as removal of paragraph numbers at the beginning of sentences). The preprocessing also included the application of an n-gram recognizer for complex references to (sections of) regulations, such as *"(EWG) Nr. 1408/71 [3]"* (see [7]). In order to simplify the parsing of more sentences involving such constructions, the training data of the section recognizer was considerably extended. The

[5] see `http://wt.jrc.it/lt/Acquis/` and `http://www.europarl.europa.eu/`; the specific fragment of JRC Acquis and Europarl used by ResPubliQA is available from `http://celct.isti.cnr.it/ResPubliQA/Downloads`.

Table 1. Parsing Rate of WOCADI on the ResPubliQA corpora

Corpus	full parse	partial parse
JRC Acquis (2009 parse)	26.2%	54.0%
JRC Acquis (new parse)	35.1%	61.5%
Europarl	34.2%	82.5%

achieved parsing rates in Table 1 show a positive effect of these changes, but both JRC Acquis and Europarl are still very hard to parse.

The pre-analysed sentences are stored in a Lucene-based retrieval system.[6] Note that instead of word forms or stems, the system indexes all possible word senses for each sentence. Moreover nominalization relationships are utilized for enriching the index. A system of 49,000 synonym classes involving 112,000 lexemes is used for normalizing synonyms, i.e. a canonical representative is chosen for all terms in a given synonym class. A special treatment of compounds was added so that all parts of the compound are indexed in addition to the compound itself. Moreover, occurrences of expected answer types (like PERSON, DATE) in each sentence are indexed. The recognition of these answer types rests on the named entity information provided by WOCADI. The presence of certain word senses and regular expressions defined on the morpho-lexical analysis of WOCADI can also trigger the recognition of these answer types. Currently there are 897 such triggers (including 704 newly added triggers for OPINION questions, and some additional triggers for the PROCEDURE type).

For definition questions, a special definition index was generated. In this index, only the definiendum of a definition recognized in a sentence is used for indexing. For example, consider this definition:

> *Hopfenpulver: Das durch Mahlen des Hopfens gewonnene Erzeugnis, das alle natürlichen Bestandteile des Hopfens enthält.* (Hop powder: the product obtained by milling the hops, containing all the natural elements thereof)

Here, only the word sense *hopfenpulver.1.1* of the defined term *Hopfenpulver* (Hop powder) is added to the definition index. This ensures a high retrieval precision for definition questions. The recognition of definitions in the texts was adjusted such as to cover the typical forms of definitions in administrative texts.

For retrieving a set of candidate sentences, the JRC Acquis index and the Europarl index are searched in parallel (using Lucene's *MultiSearcher*), and the 200 best sentences for the given query are fetched.

The retrieval query is constructed from disambiguated word senses in the question analysis if a full parse exists. Otherwise a frequency criterion is used

[6] see `http://lucene.apache.org/`

to select a unique word sense from the set of alternatives for each word in the question. Synonyms are again normalized by choosing a canonical representative. In the example, the retrieval query becomes:

> *herr.1.1 barroso.0 bei.1.1 unterzeichnung.1.1 nabucco-abkommen.1.1 nabucco.0 abkommen.1.1 sagen.1.1*

The expansion of the compound *Nabucco-Abkommen* is intended to improve recall. The retrieval query will be extended by an additional term that expresses the expected answer type, in this case *atype:OPINION*. Note that the *atype:x* term is always an optional part of the retrieval query (it can be dropped at the expense of the retrieval score). This is different from the approach in ResPubliQA 2009 where for definition questions, *atype:DEFINITION* was treated as a required subexpression so that sentences not recognized as containing a definition were completely dropped.

2.4 Extraction of Shallow Validation Features

As the basis for selecting the best retrieved sentence, several features that describe the quality of the question/snippet match are computed. We start by describing some shallow features that can be computed for arbitrary sentences regardless of the success of parsing. Apart from the obvious expected answer type check, another important method is a lexical overlap test. To this end, LogAnswer determines the list of all (synonym normalized) word senses for each word in the question (except stopwords). Each list of alternative word senses is treated as a disjunction one of whose elements must find a match in the sentence to be validated. In our runnung example, we get:

> (*herr.1.1*) (*barroso.0*) (*signieren.1.1 unterzeichnung.1.1*) (*nabucco-abkommen.1.1 nabucco.0 nabucco.1.1*) (*abkommen.1.1 nabucco-abkommen.1.1*) (*sagen.1.1 besagen.1.1 sagen.2.1*),

where the canonical synonym *signieren.1.1* replaces the original *unterzeichnen.1.1*.

A recent change to LogAnswer is the treatment of nominal compounds: Each compound is split in two conjuncts so that a full match is possible if the text either contains the compound directly (*Nabucco-Abkommen*), or alternatively, if it contains the components of the compound (i.e. both *Nabucco* and *Abkommen*).

In the example, this candidate sentence is found that answers the question:

> *Am 13. Juli bei der Unterzeichnung des Nabucco-Abkommens in Ankara sagte Herr Barroso, die Gas-Pipelines seien aus Stahl.* (On 13 July in Ankara, at the time of signing the Nabucco agreement, Mr Barroso said that the gas pipelines were made from steel.)

LogAnswer then extracts the following (synonym normalized) word senses and numerals from the morpho-lexical analysis of this sentence:

> *abkommen.1.1*, *ankara.0*, *barroso.0*, *familiename.1.1*, *gas.1.1*, *gaspipeline.1.1*, *herr.1.1*, *monat.1.1*, *nabucco-abkommen.1.1*, *name.1.1*, *past.0*, *pipeline.1.1*, *present.0*, *sagen.1.1*, *sein.3.8*, *stadt.1.1*, *stahl.1.1*, *tag.1.1*, *unterzeichnung.1.1*, *7*, *13*.

Here, every list of alternative word senses from the question representation finds a match in the shallow sentence representation. Thus, there are 0 cases of matching failure (100% matching rate). For details on the shallow features, see e.g. [3].

2.5 Extraction of Logic-Based Validation Features

For sentences with a full parse, a relaxation proof of the (logical representation of the) question from the logical representation of the sentence and the available background knowledge is also tried, resulting in additional logic-based features. The prover works on synonym-normalized representations. The background knowledge comprises more than 10,500 facts (e.g. describing nominalizations), and 114 rules for basic inferences, see e.g. [4].

Recalling our running example, the following logical query is constructed, based on the question parse and the result of question classification:

$$\begin{aligned}&\mathrm{attr}(X_1, X_2), \mathrm{sub}(X_1, \textit{herr.1.1}), \mathrm{val}(X_2, \textit{barroso.0}), \mathrm{sub}(X_2, \textit{familiename.1.1}),\\&\mathrm{obj}(X_3, X_4), \mathrm{subs}(X_3, \textit{unterzeichnung.1.1}), \mathrm{sub}(X_4, \textit{nabucco-abkommen.1.1}),\\&\mathrm{agt}(X_5, X_1), \mathbf{circ}(X_5, X_3), \mathrm{subs}(X_5, \textit{sagen.1.1}), \mathrm{mcont}(X_5, F)\end{aligned}$$

All variables are assumed to be existentially quantified. Comma means conjunction. The variable F is the question focus (it expresses the queried information).

The logical representation of the correct answer sentence shown above is:

$$\begin{aligned}&\mathrm{val}(c_{10}, c_7), \mathrm{sub}(c_{10}, \textit{monat.1.1}), \mathrm{obj}(c_{14}, c_{17}), \mathrm{subs}(c_{14}, \textit{unterzeichnung.1.1}),\\&\mathrm{loc}(c_{17}, c_{262}), \mathrm{sub}(c_{17}, \textit{nabucco-abkommen.1.1}), \mathrm{origm}(c_{180}, c_{257}),\\&\mathrm{pred}(c_{180}, \textit{gaspipeline.1.1}), \mathrm{arg1}(c_{182}, c_{180}), \mathrm{arg2}(c_{182}, c_{257}), \mathrm{temp}(c_{182}, \textit{present.0}),\\&\mathrm{subs}(c_{182}, \textit{sein.3.8}), \mathbf{assoc}(c_{22}, c_{14}), \mathrm{mcont}(c_{22}, c_{182}), \mathrm{agt}(c_{22}, c_{31}), \mathrm{temp}(c_{22}, c_8),\\&\mathrm{temp}(c_{22}, \textit{past.0}), \mathrm{subs}(c_{22}, \textit{sagen.1.1}), \mathrm{attr}(c_{24}, c_{25}), \mathrm{sub}(c_{24}, \textit{stadt.1.1}),\\&\mathrm{val}(c_{25}, \textit{ankara.0}), \mathrm{sub}(c_{25}, \textit{name.1.1}), \mathrm{sub}(c_{257}, \textit{stahl.1.1}), \mathrm{in}(c_{262}, c_{24}),\\&\mathrm{attr}(c_{31}, c_{32}), \mathrm{sub}(c_{31}, \textit{herr.1.1}), \mathrm{val}(c_{32}, \textit{barroso.0}), \mathrm{sub}(c_{32}, \textit{familienname.1.1}),\\&\mathrm{attr}(c_8, c_{10}), \mathrm{attr}(c_8, c_9), \mathrm{val}(c_9, c_6), \mathrm{sub}(c_9, \textit{tag.1.1}), \mathrm{assoc}(\textit{gaspipeline.1.1}, \textit{gas.1.1}),\\&\mathrm{sub}(\textit{gaspipeline.1.1}, \textit{pipeline.1.1}), \mathrm{sub}(\textit{nabucco-abkommen.1.1}, \textit{abkommen.1.1})\end{aligned}$$

Due to a parsing error, the logical query shown above cannot be proved from the representation of the sentence – the parser has used an unspecific **assoc** relation instead of the **circ** relation in the query. Thus, the critical literal will be skipped from the query, resulting in a proof of the remaining query fragment.

Among the logic-based features that summarize the relaxation proof, there is one feature reporting that a single literal had to be skipped, and another feature reporting that $10/11 \approx 91\%$ of the query literals have been proved. For details on these logic-based features and the use of relaxation in LogAnswer, see [3].

2.6 Selection of the best answer candidate

The selection of the best answer candidate is based on the shallow features obtained by matching the question and the sentence terms, and (for sentences with a full parse) also on features obtained from a relaxation proof of the question from the candidate sentence. Rank-optimizing decision trees [5] are used for assigning a validation score to each retrieved sentence that allows the selection of the best candidate. For the moment, LogAnswer still uses a validation model based on annotated training data from the QA@CLEF 2007 and 2008 evaluations. Using the ResPubliQA 2009 test set for preparing training data seemed too complicated for a non-expert of EU legislation.

The c@1 evaluation metric of ResPubliQA rewards QA systems that validate their answers and prefer not answering over wrong answers. Thus results with a low validation score should be dropped since their probability of being correct is so low that showing these results would reduce the c@1 score of LogAnswer. The threshold $\theta = 0.09$ for accepting the best answer, or generating a 'NO ANSWER' response if the validation score falls below the threshold, was chosen such as to optimize the c@1 score of LogAnswer on the ResPubliQA 2009 test set.

Finally, if the best sentence is not rejected, then it is expanded to the corresponding full paragraph, as required by the ResPubliQA PS task.

2.7 Reasoning Support by the E-KRHyper Theorem Prover

E-KRHyper [11] was used as the reasoning component. E-KRHyper is an automated theorem prover (ATP) for first-order logic with equality. It is based on an extended form of the hyper tableaux calculus [1,2]. The system is implemented in OCaml[7], and it is available under the Gnu GPL from the E-KRHyper website[8].

While we developed this prover for embedding in knowledge representation applications, it can operate as a stand-alone theorem prover which accepts input problems in the syntax used by the TPTP logic problem library [16], a standard in automated theorem proving. The TPTP website[9] provides a periodically updated performance listing of a number of ATP systems with respect to the problem library. At the time of this writing E-KRHyper solves 26% of the problems. In comparison the *Otter* [10] system solves 19%; Otter serves as a benchmark in ATP testing due to its long history and stability. While leading ATP systems like *E* [15] and *Vampire* [14] exceed 50% in the TPTP rankings, E-KRHyper is very suitable to the type of logic problems arising in knowledge representation and question answering, characterized by a large number of clauses, of which only a select few are actually necessary for the eventual proof. Regarding this problem class our system generally outperforms other theorem provers [3]. The QA-oriented logic problems used in our tests were derived from computations in previous CLEF competitions. We have since submitted a selection of about 200 of these problems to the TPTP, and they have been included in the problem library in the CSR domain (common sense reasoning) as of TPTP v4.0.1.

E-KRHyper has several features which support its role as a reasoning server within an application like LogAnswer. Logic extensions like equational reasoning,

[7] `caml.inria.fr`

[8] `http://www.uni-koblenz.de/~bpelzer/ekrhyper`

[9] `www.tptp.org`

Table 2. Results of LogAnswer in ResPubliQA 2010. #right cand. is the number of correct paragraphs at top rank before applying θ, and accuracy = #right cand./#questions

run	description	#right cand.	accuracy	c@1 score
loga101PSdede	standard system	103	0.52	0.59
loga102PSdede	special def index	107	0.54	0.62

negation as failure, arithmetic evaluation and list processing enable the prover to handle aspects of knowledge representation systems which cannot be expressed within the limits of first-order logic. Most ATP systems are designed to work on a single problem only. They terminate once they have found a result, and they must be started anew for each problem. This mode of operation would be impractical for a reasoning server within LogAnswer, as it would entail reloading the extensive logical knowledge base for each query, every time rebuilding the indexing structures which the prover requires for fast clause access. Instead E-KRHyper can remain in operation indefinitely, loading the knowledge base only once during the initialization of LogAnswer. Any additional clauses required by the queries can be loaded and retracted during the operation.

For query relaxation E-KRHyper supplies LogAnswer with information about partially successful proof attempts. LogAnswer selects the most promising way to relax the query, and then the prover continues with the shortened query, keeping any previous derivation results to avoid repeating inferences.

When E-KRHyper finds a proof, it extracts the answer substitution from the proof and transfers it to the main LogAnswer system for further processing.

3 Results on the ResPubliQA 2010 Test Set for German

Two runs were produced for ResPubliQA 2010: The first run, *loga101PSdede*, represents the LogAnswer system in its standard configuration without the experimental definition index. In *loga102PSdede*, by contrast, the definition index was activated. While the system was configured to retrieve 200 candidate sentences for each question, the actual number of available sentences was smaller in some cases. In the *loga101PSdede* run, a total of 39,200 candidate sentences was retrieved (196 per question). About 37.0% of the retrieved candidate sentences had a full parse, thus allowing logical validation. The remaining candidate sentences with a chunk parse (41.9%) or failed parse (21.1%) were only subjected to a shallow validation (numbers for the *loga102PSdede* run are similar). The results obtained for the two submitted runs are shown in Table 2. The use of the definition index in *loga102PSdede* was a clear improvement.

Table 3 shows the results of corresponding shallow-only runs (generated with the prover switched off), and the results of two IR baseline runs in which the top-ranked sentence of the retrieval stage was directly used for choosing the corresponding answer paragraph. Note that a different 'no answer' threshold of

Table 3. Ablation Results for LogAnswer: The shallow-only runs were performed with the prover switched off. The IR baseline runs simply return the best retrieved candidate.

run	description	#right cand.	accuracy	c@1 score
SH-101	shallow-only validation	105	0.53	0.60
SH-102	shallow-only, def index	108	0.54	0.62
IR-101	IR baseline	82	0.41	0.43
IR-102	IR baseline, def index	88	0.44	0.47

Table 4. Accuracy by question category. The runs only differ for definitions

Run	REAS/PURP	FACTOID	PROC	OPINION	OTHER	DEFINITION
	(33)	(35)	(33)	(33)	(34)	(32)
loga101PSdede	0.70	0.66	0.52	0.45	0.41	0.34
loga102PSdede	0.70	0.66	0.52	0.45	0.41	0.41

$\theta = 0.76$ was used for the IR baseline runs. In this case the treshold was used for cutting off results with a poor Lucene retrieval score. The value of $\theta = 0.76$ was again chosen such as to optimize the corresponding c@1 score on the ResPubliQA 2009 questions. The table clearly shows that the use of logical validation techniques provided no extra benefit in the ResPubliQA task – the results of the standard system (with logical validation) and of the configuration that only uses shallow features for validation was about the same. Comparing the detailed results of the runs of the full system (*loga101PSdede*) and the shallow only configuration (*SH-101*), we found that the validation models resulted in a different choice of shown answer only for 15% of the questions. For *loga102PSdede* vs. *SH-102*, the chosen answer paragraphs were different for 14% of the questions. Given the high accuracy achieved by the shallow validation technique, logical validation was obviously not called for by the ResPubliQA task. An interesting finding was that (with one exception), all questions for which deep validation outperformed the shallow-only technique were DEFINITION questions.

On the other hand, the validation techniques of LogAnswer achieved a strong benefit compared to using the retrieval score only. Comparing *loga101PSdede* and the *IR-101* run, for example, accuracy increases by 27% and the c@1 score increases by 37% due the use of validation instead of the plain retrieval result.

A breakdown of results by question category is shown in Table 4. LogAnswer was best for REASON/PURPOSE and FACTOID questions. DEFINITION and OTHER questions performed worst. OPINION questions also proved difficult. The use of a special definition index in the second LogAnswer run increased the accuracy for definition questions from 34% to 41% compared to the first run with a single index for all questions. This result is encouraging, though more work has to be done here. Compared to the results of LogAnswer in ResPubliQA 2009 [7],

Table 5. Success rate of question classification

Category	#questions	#recognized	recog-rate	(last year)
REAS/PURP	33	30	90.9%	70.1%
FACTOID	35	31	88.6%	70.5%
DEFINITION	32	28	87.5%	85.3%
OTHER	34	29	85.3%	–
PROCEDURE	33	26	78.8%	20.3%
OPINION	33	19	57.8%	–
(total)	200	163	81.5%	65.6%

there was a strong improvement for all question types. PROCEDURE questions (shown as PROC in the table) are no longer problematic for LogAnswer.

Results on the success rate of question classification are shown in Table 5. Despite the high overall recognition rate (81.5%), the novel class of OPINION questions was obviously not yet well covered by the classification rules. Moreover, some more ways of expressing PROCEDURE questions should be supported.

The threshold $\theta = 0.09$ on validation quality that serves for cutting of poor answers was chosen such as to maximize the c@1 score of LogAnswer on last year's ResPubliQA question set. In retrospect, the optimal threshold for *loga101PSdede* would have been $\theta = 0.11$, resulting in a c@1 score of 0.60 (overall accuracy 0.52). For *loga102PSdede*, the optimal threshold would have been $\theta = 0.13$, yielding a c@1 score of 0.63 (overall accuracy 0.54). The closeness of these optimal values to the results obtained using $\theta = 0.09$ demonstrates that the method for determining the NOA threshold was effective.

We have determined the reason of failure for a sample of questions with wrong answers in the LogAnswer runs. This analysis has shown that the Lucene scoring metric in the retrieval stage of LogAnswer has a strong bias to short sentences. A short sentence that contains only one term from the IR query is often prefered to a longer sentence that contains all query terms. We thus agree with Pérez at al [12] that switching to a BM25 ranking function makes sense.

4 Conclusion

In ResPubliQA 2010, LogAnswer scored much better than in the last year. The improved accuracy for all question types shows that the general improvements of LogAnswer and the thorough utilization of compound decompositions were effective. Specifically, PROCEDURE questions are no longer problematic for LogAnswer. Compared to a system configuration with a single index, the accuracy for definition questions was increased by 21% by adding a specialized definition index. The large difference between the c@1 and accuracy scores of LogAnswer indicates a good validation performance.

Due to the low parsing rate for JRC Acquis and Europarl, we were unable to demonstrate a benefit of logical processing on the quality of results. However, as witnessed by the results of the shallow-only matching technique, the ResPubliQA task did not seem to call for sophisticated logic-based validation either.

The deep linguistic analysis of LogAnswer and its validation techniques made a substantial difference compared to a pure IR approach. Using the retrieval stage of LogAnswer as the IR baseline, we found a 27% gain in accuracy and 37% gain in c@1 due to the powerful validation techniques of LogAnswer.

References

1. Baumgartner, P., Furbach, U., Niemelä, I.: Hyper Tableaux. In: JELIA'96, Proceedings. pp. 1–17 (1996)
2. Baumgartner, P., Furbach, U., Pelzer, B.: Hyper Tableaux with Equality. In: Automated Deduction - CADE-21, Proceedings (2007)
3. Furbach, U., Glöckner, I., Pelzer, B.: An application of automated reasoning in natural language question answering. AI Communications 23(2-3), 241–265 (2010), PAAR Special Issue
4. Glöckner, I.: Filtering and fusion of question-answering streams by robust textual inference. In: Proceedings of KRAQ'07. Hyderabad, India (2007)
5. Glöckner, I.: Finding answer passages with rank optimizing decision trees. In: Proc. of the Eighth International Conference on Machine Learning and Applications (ICMLA-09). pp. 208–214. IEEE Press (2009)
6. Glöckner, I., Pelzer, B.: Combining logic and machine learning for answering questions. In: Peters et al. [13], pp. 401–408
7. Glöckner, I., Pelzer, B.: The LogAnswer project at CLEF 2009. In: Results of the CLEF 2009 Cross-Language System Evaluation Campaign, Working Notes for the CLEF 2009 Workshop. Corfu, Greece (Sep 2009)
8. Hartrumpf, S.: Hybrid Disambiguation in Natural Language Analysis. Der Andere Verlag, Osnabrück, Germany (2003)
9. Helbig, H.: Knowledge Representation and the Semantics of Natural Language. Springer (2006)
10. McCune, W.: OTTER 3.3 Reference Manual. Argonne National Laboratory, Argonne, Illinois (2003)
11. Pelzer, B., Wernhard, C.: System Description: E-KRHyper. In: Automated Deduction - CADE-21, Proceedings. pp. 508–513 (2007)
12. Pérez, J., Garrido, G., Rodrigo, A., Araujo, L., Peñas, A.: Information retrieval baselines for the respubliqa task. In: Results of the CLEF 2009 Cross-Language System Evaluation Campaign, Working Notes for the CLEF 2009 Workshop. Corfu, Greece (Sep 2009)
13. Peters, C., Deselaers, T., Ferro, N., Gonzalo, J., Jones, G., Kurimo, M., Mandl, T., Peñas, A., Petras, V. (eds.): Evaluating Systems for Multilingual and Multimodal Information Access: 9th Workshop of the Cross-Language Evaluation Forum, CLEF 2008, Aarhus, Denmark, September 17–19, Revised Selected Papers. LNCS, Springer, Heidelberg (2009)
14. Riazanov, A., Voronkov, A.: The design and implementation of Vampire. AI Communications 15(2-3), 91–110 (2002)
15. Schulz, S.: E - a brainiac theorem prover. AI Communications 15(2-3), 111–126 (2002)
16. Sutcliffe, G., Suttner, C.: The TPTP Problem Library: CNF Release v1.2.1. Journal of Automated Reasoning 21(2), 177–203 (1998)