The LogAnswer Project at QA4MRE 2012

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`

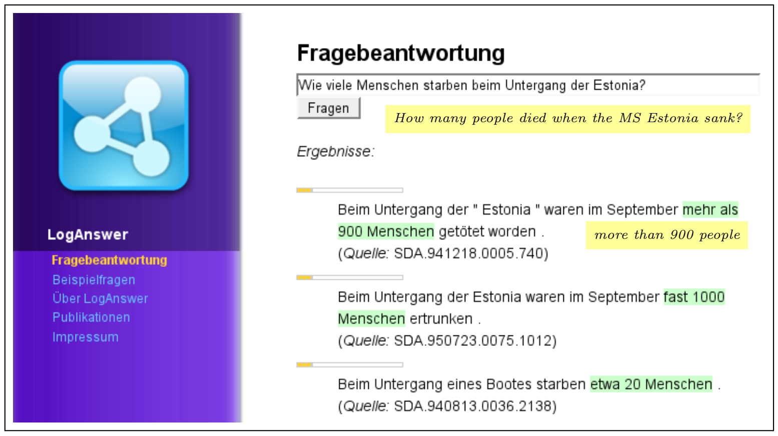
[2] Department of Computer Science, Artificial Intelligence Research Group University of Koblenz-Landau, Universitätsstr. 1, 56070 Koblenz
`bpelzer@uni-koblenz.de`

Abstract. For QA4MRE 2012, we have improved our prototype of a pure logic-based answer validation system that was originally developed for QA4MRE 2011. The system uses core functionality of the LogAnswer question answering (QA) system available on the web, which was combined and extended such as to best meet the demands of the QA4MRE task. In particular, coreference resolution is used in order to allow knowledge processing on the document level, and a fragment of OpenCyc has been integrated in order to allow more flexible reasoning. In addition to a general consolidation of the original prototype that took part in QA4MRE 2011, the current system was optimized for accuracy on non-rejected questions. While last year's system only achieved an accuracy of 0.24 for answered questions, the various measures taken to improve the rejection decision now yield an accuracy of 0.31 for answered questions in the QA4MRE 2012 task. Results show that these improvements were sufficient to raise the overall quality of answers to a c@1 score of 0.26, which is clearly above the c@1 baseline of 0.20 for random guessing. Due to the difficulty of the reading tests in QA4MRE, we consider this result promising, recalling the near-baseline scores of most QA4MRE participants in 2011.[1]

1 Introduction

The DFG-funded project *LogAnswer* investigates the use of deep linguistic processing and logical reasoning for QA, with a special emphasis on the issues of robustness and efficiency. An experimental QA system for German (also called LogAnswer) [2] that evolved from this research can be tested online (`www.loganswer.de`), a screenshot is shown in see Fig. 1. The system took part in the QA@CLEF and ResPubliQA QA system evaluations [3,4,5]. Recently, we have focused on the issue of providing automatic QA support for human Q&A portals. Users of the German FragWikia! portal (`http://frag.wikia.com/`) can subscribe to the automatic answer service on the LogAnswer website. LogAnswer keeps track of new questions at FragWikia! and sends an answer email whenever

[1] LogAnswer is supported by the DFG with grants FU 263/12-2 and GL 682/1-2.



Fragebeantwortung

Wie viele Menschen starben beim Untergang der Estonia?

How many people died when the MS Estonia sank?

Ergebnisse:

Beim Untergang der " Estonia " waren im September mehr als 900 Menschen getötet worden . *more than 900 people*
(*Quelle:* SDA.941218.0005.740)

Beim Untergang der Estonia waren im September fast 1000 Menschen ertrunken .
(*Quelle:* SDA.950723.0075.1012)

Beim Untergang eines Bootes starben etwa 20 Menschen .
(*Quelle:* SDA.940813.0036.2138)

Fig. 1. Screenshot of the LogAnswer QA system, using QA@CLEF data

it finds sufficiently good results for a question of a registered user. In this new use case for QA, we face the challenge that there is often few lexical overlap between question and answer. Moreover the system must validate answers in order to achieve better precision. This is because users do not necessarily expect the system to answer at all in this scenario. But if it does post a response, then the answer should show a high probability of correctness, in order to avoid users getting annoyed by the machine-generated answers. This scenario resembles the present QA4MRE machine reading exercise [6] in that there is often no close relationship between question and answers, and in the need to validate answers rather than always providing a ranked list of answer candidates.

Despite these similarities that make the QA4MRE participating appealing for our project, application of existing LogAnswer system (even with the precision-oriented refinements for QA portals) to the QA4MRE task does not make sense:

– LogAnswer normally finds answer from scratch, based on a candidate retrieval backend whose retrieval scores are considered in determining the final ranking. In QA4MRE, however, there is no retrieval score as one of the starting points for reranking.
– LogAnswer usually works with a sentence-level segmentation of documents, while QA4MRE may demand a justification of answers from several parts of a document or even from several documents.
– Since the validation features of LogAnswer depend on sentence-level segmentation and since the retrieval score as one of these features, the validation models of LogAnswer as determined from a learning-to-rank approach are not applicable.
– The existing QA@CLEF-based training set for the machine learning (ML) technique cannot be reused either because it shows strong effect of lexical

overlap (unlike QA4MRE) and because it comprises sentence-level items only.

Owing to these difficulties that were already present in QA4MRE 2011, we decided to build a new LogAnswer prototype for the QA4MRE task in the last year [7], using only those basic techniques from the regular LogAnswer system that seemed useful for the task:

– The core system for QA4MRE was designed to achieve logical answer validation on document level representations;
– Existing LogAnswer functionality was reused whenever possible;
– Instead of using the supplied "QA4MRE Background Collection" or a web-based validation approach, we decided to strengthen the background knowledge of LogAnswer by integrating part of OpenCyc;
– Generation of explanations for a chosen answer was added.

For QA4MRE 2012, our goal was a general consolidation of the LogAnswer prototype built for QA4MRE and an improvement of its capabilities for rejecting wrong answers in particular.

2 System Description

The LogAnswer prototype for QA4MRE relies on a deep linguistic analysis.

– The reading tests, questions and answer candidates are parsed by the WOCADI parser [8].
– WOCADI computes a meaning representation for each sentence in the form of a Multilayered Extended Semantic Network (MultiNet [9]).
– The question is then subjected to a question classification, based on a system of 248 classification rules that operate on the generated semantics networks.

In QA4MRE 2011, the question classification was of minor importance compared to regular Q&A here since no predictive annotation and no expected answer type check was used. For QA4MRE 2012, some effort was spent in utilizing the computed question category, specific answer types (as known from the former QA@CLEF and ResPubliQA exercises) and other information determined by the question classification rules in order to eliminate questions that the system is unable to answer:

– LogAnswer is designed for answering single-sentence questions. Therefore, questions that span more than one sentence are now discarded.
– LogAnswer has no special means for validating list answers (e.g. answers involving a coordination such as 'London and Paris'). Therefore question that target at several answers (such as 'Which cities. . . ') are now recognized and discarded since they are not likely to be answered correctly.
– Questions are now generally discarded if no expected answer type can be established, except if the question is classified into the 'DEFINITION' category (for definition questions).

In the following we sketch the construction of the logical representation on the document level.

- *Coreference resolution* is essential to the success of logic-based processing on whole documents: under the unique name assumption, the combined information about a discourse entity can only be utilized if all mentions of the entity in the text are represented by the same constant. We therefore use a coreference resolver, CORUDIS [8], for handling anaphoric pronouns and other forms of coreference.
- The coreference resolver is complemented with a simplistic solution to *identifying the speaker* from the manuscript of a talk which was activated in the QA4MRE runs. The method simply chooses the first name mentioned at the beginning of the manuscript. Knowing the speaker is necessary for resolving first person deixis *'Ich'/'I'*).
- In a process called *assimilation*, the text representation is constructed from all logical sentence representations, thereby using the results of coreference analysis and speaker recognition.

In order to capture more inferences, ca. 20,000 subsumption relationships were extracted from OpenCyc and translated into our concept system for German, see Section 3. For efficiency reasons, this subsumption hierarchy is not added to the background knowledge seen by the prover. Instead, the text representation is enriched by explicitly declaring each instance of a concept an instance of all of its superconcepts as well.

Example:

- A document mentions a tsunami.
- Logical representation contains statement like $\mathrm{SUB}(\mathit{c24}, \mathit{tsunami.1.1})$, expressing that entity *c24* is an instance of a tsunami (with HaGenLex word sense index *1.1*, see [10]).
- From the OpenCyc link, we know that every tsunami is a 'Naturkatastrophe' (natural disaster).
- The OpenCyc expansion step then adds an assertion expressing that *c24* is also an instance of a natural disaster, viz $\mathrm{SUB}(\mathit{c24}, \mathit{naturkatastrophe.1.1})$.

Let us now turn to *hypothesis construction*. By a *hypothesis*, we mean a declarative statement formed from the question and the candidate answer. Consider the question *'What is Bono's attitude with respect to the digital age?'* and the possible answer *"enthusiasm"*. In this case, the textual hypothesis would be *'Bono's attitude with respect to the digital age is enthusiasm'*. In our prototype, the template-based construction of a textual hypothesis from question and answer, followed by parsing, is only used as a fallback. This is because the text-based hypothesis construction is problematic for German due to mismatches of word forms with respect to syntactic case. Instead, our system normally constructs the logical hypothesis directly on the logical level, by combining the literal lists that represent question and answer.

Based on the logical hypothesis (as a list of literals) and the logical representation of the document for the given reading test, our system then uses *robust logical processing* for justifying or rejecting the hypothesis (and in turn, the answer candidate). This procedure follows the general principle of logical answer validation:

– Try to prove the logical hypothesis from the logical representation of the reading test document and from the general background knowledge of the system.
– If such a proof succeeds, then the answer from which the hypothesis was constructed is considered correct.

In practice, linguistic analyses contain errors, coreference resolution can fail, and the background knowledge is incomplete. Therefore, the proof attempt may fail for correct hypotheses, and we cannot conclude from a failed proof that the hypothesis is indeed wrong. For ensuring sufficient robustness, we thus add a relaxation loop to the logical validation process:

– Remove problematic hypothesis literals one at a time until a proof of the remaining hypothesis fragment succeeds.
– Impose a limit on the number of admissible relaxation steps in order to keep processing time within reasonable bounds.

The skipped literal count and other criteria related to the relaxation result are then used as features that affect answer scoring.

QA4MRE 2011 required systems to provide one or more so-called provenance elements that justify the chosen answer. This requirement was simplified in this year's task, which only assumes general information of the resources involved to be provided. Still, we did not eliminate the extraction of detailled justifications from our system since it proved useful for computing validation scores, depending on the coherence of found justifications.

The computation of provenance information (i.e. of the individual justifying elements that together explain the answer) is based on prover results:

– All names of used axioms (implicative rules) are added as provenance elements, since axiom names in our system are chosen in a meaningful way.
– Used facts from the background knowledge generally express lexical-semantic relations, such as CHPA(*hoch.1.1*, *höhe.1.1*) (i.e., property 'high' corresponds to attribute 'height'). Knowing the MultiNet documentation, such a fact is self-explanatory and can be added as a provenance element.
– For literals from the text representation such as SUB(*c24*, *tsunami.1.1*), the system presents a witness sentence rather than the used fact itself. These witness sentences are also considered as provenence elements and thus included into the computed justification.

Since no sufficient training data was available for applying a machine-learning technique that automatically determines validation scores, we used an ad-hoc scoring metric that closely resembles the validation criterion already used in the last year.[2] Many of the validation features normally used by LogAnswer had to be omitted since it was not clear how to utilize them in a simple hand-crafted scoring criterion.

As in the last year, we started from a basic scoring metric ϱ defined as the arithmetic mean of:

- $\varrho_1 = \alpha^{\#\textit{skipped-lits}}$, where $\alpha = 0.7$.[3]
- $\varrho_2 = \alpha^{\#\textit{skipped-lits}} \cdot \beta^{\#\textit{unknown-status-lits}}$, where $\beta = 0.8$.[4]
- ϱ_3: optimistic proportion of provable question literals:

$$\varrho_3 = 1 - \frac{\#\textit{skipped-q-lits}}{\#\textit{all-q-lits}}$$

- ϱ_4: pessimistic proportion of proved question literals:

$$\varrho_4 = \frac{\#\textit{proved-q-lits}}{\#\textit{all-q-lits}}$$

- ϱ_5: optimistic proportion of provable answer literals:

$$\varrho_5 = 1 - \frac{\#\textit{skipped-a-lits}}{\#\textit{all-a-lits}}$$

- ϱ_6: pessimistic proportion of proved answer literals:

$$\varrho_6 = \frac{\#\textit{proved-a-lits}}{\#\textit{all-a-lits}}$$

This basic validation score is then combined with a criterion intended to capture the coherence of the justification of the answer as found by the system using a relaxation proof. In order to determine this coherence score, we organize the witness sentences into blocks of adjacent sentences. Let us say that a block of adjacent witness sentences is 'unconnected' to earlier blocks if it is not linked by coreference to these blocks. Let u be the number of unconnected blocks of witness sentences for the given answer a. The *final answer selection score* σ is then computed from the basic score ϱ and the unconnected sentence score u as

$$\sigma = \varrho \cdot \gamma^{u-1}$$

where $\gamma = 0.7$ in our experiments.

The final step in solving an individual reading test item is the selection of the best answer, or alternatively the decision not to commit to any answer at all, leaving the question unanswered.

[2] The QA4MRE 2011 data for LogAnswer was considered too small for learning a useful model.

[3] *#skipped-lits* is the number of skipped hypothesis literals due to relaxation.

[4] *#unknown-status-lits* is the number of non-skipped hypothesis literals not yet proved in the last proof attempt of the relaxation loop.

– The prototype generally refuses to answer if question analysis fails (i.e., if there is no full parse of the question or an empty question literal pattern).
– An answer candidate is also rejected if hypothesis construction fails (e.g. if the constructed hypothesis contains no literals associated with the original answer).

As opposed to our system setup in QA4MRE 2011 described in [7], the prototype no longer rejects answer candidates for which provenance generation fails. In this case that an empty justification for the answer is extracted, the system rather determines σ based on $u = 1$ (i.e. assuming that the candidate answer is justified by a single witness sentence).

If all answers are rejected for one of the reasons mentioned, then the question is left unanswered. Otherwise the answer with maximum score is chosen.

3 Resources Used

The background knowledge of LogAnswer comprises:

– A system of 49,000 synonym classes involving 112,000 lexemes, used for synonym normalization (concepts are replaced by a canonical representative);
– About 10,500 facts, e.g.
  – nominalization relationships ('to attack' vs. 'the attack')
  – adjective-attribute relationships ('high' vs. 'height')
  – gender-specific profession names ('Professor' vs. 'Professorin' [female professor] in German)
– Approx. 120 rules for basic inferences (e.g. for making use of the lexical-semantic relationships listed above).

In order to enhance the background knowledge, the system is further equipped with knowledge from OpenCyc (`http://sw.opencyc.org/`), To this end, the HaGenLex [10] concept system of German word senses was linked to a fragment of OpenCyc using a semi-automatic alignment technique. We started with automatic generation of hypotheses for HaGenLex-OpenCyc links:

– HaGenLex $\rightarrow$ wikipedia.de topic or redirect $\rightarrow$ DBPedia map (de-en) $\rightarrow$ wikipedia.en-OpenCyc link, or alternatively:
– HaGenLex $\rightarrow$ HaEnLex $\rightarrow$ wikipedia.en topic/redirect $\rightarrow$ wikipedia.en-OpenCyc link,

where HaEnLex is a core lexicon for English designed analogously to HaGenLex and interlinked with the German lexicon. The automatic generation of link hypotheses was followed by a manual validation, checking several ten thousands of links generated in this way for their actual correctness. The checked links are then used for knowledge extraction: the validated alignment lets us map synonyms, subsumptions, and disjoint declarations from OpenCyc into our HaGenLex concept system. A final refinement consists in a check for cyclic subsumptions and inconsistencies, which are found automatically but had to be fixed by hand.

Table 1. Main results of LogAnswer prototype in QA4MRE 2012

Run	aw	corr	acc|aw	acc	c@1
loga11011dede	96	28	0.29	0.18	0.25
loga12023dede	96	30	0.31	0.19	0.26

Moreover, there is an automatic elimination of redundancy (i.e., optimization of subsumption chains). This process resulted in 20,000 new redundancy-free subsumption relationships, 1,500 new synonyms, and 4,000 disjointness declarations, which substantially extend the background knowledge available to LogAnswer.

4 Experimental Results

Two runs were submitted to QA4MRE 2012, using German as both the source and target language. These runs were based on the following system configurations:

– *loga12011dede*: only elementary synonyms (spelling variants, wrong spellings) and elementary background knowledge (e.g. simple temporal and spatial regularities), no OpenCyc integration.
– *loga12023dede*: full background knowledge of LogAnswer including OpenCyc expansion.

The main evaluation results obtained for the 160 test questions of QA4MRE 2012 are shown in Table 1. Here, 'aw' is the number of answered (non-rejected) questions, 'corr' is the number of answered questions with a correct choice of the answer, 'acc|aw' is the accuracy achieved for answered questions, 'acc' is overall accuracy, and 'c@1' is the c@1 score achieved by the given run. The system attempts to answer 94 questions, with 28 correct answers in the first run and 30 correct answers in the second run supported by the full background knowledge of LogAnswer. Thus, there is a slight positive effect of the added knowledge including the OpenCyc integration. The achieved c@1 score of 0.26 clearly exceeds the c@1 score of 0.20 for random guessing. The accuracy of 0.31 for non-requected questions in the second run is substantially higher than the corresponding value of 0.24 achieved in the last year [7]. Moreover the c@1 score has improved from 0.22 to the current 0.26. This indicates a positive effect of the consolidation of the system and of the refinements for recognizing questions that the system is unable to handle.

Table 2 also shows the results by topics. T1. T2, T3 and T4 are the c@1 scores achieved for topics 1 (AIDS), 2 (Climate Change), 3 (Music and Society), and 4 (Alzheimer), respectively. 'all' is the overall c@1 score for comparison.

While the self-assessment of the system with respect to failed answer attempts has been improved, mainly by better exploiting the results of question classification, some problems of the prototype with the QA4MRE tasks remain:

Table 2. Topic results of LogAnswer prototype in QA4MRE 2012

Run	T1	T2	T3	T4	all
loga11011dede	0.26	0.07	0.42	0.23	0.25
loga12023dede	0.29	0.11	0.42	0.23	0.26

- *Poor document quality*
  One of the main problem for deep linguistic processing were missing blanks, resulting in adjacent words to be wrongly merged into single tokens. There was also no structuring of the documents by blank lines at all, so that headlines were not clearly separated from regular text. Due to parsing failure, this resulted in many sentences to become invisible for logic-based validation.
- *Spelling errors in questions*
  The WOCADI parser has found a full parse for 132 out of the 160 questions. Parsing problems in the remaining 28 questions were mostly due to spelling errors or wrong grammar. These problems made it impossible for the system to answer the corrupted questions.
- *Linguistic diversity of answers*
  Due to the rich question types in the reading tests, answers were syntactically diverse and hard to parse (e.g. incomplete sentences).
- *Missing background knowledge*
  The regularities needed to understand the correctness of an answer were complex and hard to capture by logical rules.

5 Conclusions and Future Work

We have described the LogAnswer prototype for QA4MRE 2012, which resulted from last year's system by a general consolidation and refinements for better recognizing questions that the system cannot answer. These improvements have raised the c@1 score of the system to 0.26, which is substantially better than the expected c@1 score of 0.20 for random guessing. There was also a slight positive effect of using background knowledge including the OpenCyc integration, compared to using no background knowledge at all. However, defects in reading test documents (missing blanks and structuring) still make it difficult to apply deep linguistic processing and subsequent logical reasoning for validating answers. No 'shallow' techniques were used as a fallback if the deep linguistic analysis fails, since lexical overlap and similar shallow linguistic criteria will not help too much in QA4MRE. It would have been best to complement logical answer validation with web-based validation or hit scores obtained from the QA4MRE Background Collection. The modest c@1 score achieved by LogAnswer can also be attributed to the fact that the QA4MRE background collections were not yet used by the system. Looking at concrete questions, a decomposition of coordinating answers such as 'London and Paris' would have been useful, but for the time being the system discards questions that target at such answers. From a more general perspective, we will continue working on techniques for increasing the precision of results in order to make our system more useful in the QA portals use case. One concrete approach that we are currently working on is the use of techniques from Case-Based Reasoning (CBR). These techniques could help to utilize user feedback for continually improving validation quality.

References

1. 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, Revised Selected Papers, Heidelberg, Springer (2009)
2. Furbach, U., Glöckner, I., Pelzer, B.: An application of automated reasoning in natural language question answering. AI Communications **23**(2-3) (2010) 241–265
3. Glöckner, I., Pelzer, B.: Combining logic and machine learning for answering questions. [1]
4. Glöckner, I., Pelzer, B.: The LogAnswer project at CLEF 2009. In: Working Notes for the CLEF 2009 Workshop, Corfu, Greece (September 2009)
5. Glöckner, I., Pelzer, B.: The LogAnswer project at ResPubliQA 2010. In: CLEF 2010 Working Notes. (September 2010)
6. Peñas, A., Hovy, E., Forner, P., Rodrigo, A., Sutcliffe, R., Sporleder, C., Forascu, C., Benajiba, Y., Osenova, P.: Overview of QA4MRE at CLEF 2012: Question Answering for Machine Reading Evaluation. In: CLEF 2012 Evaluation Labs and Workshop Working Notes Papers, Rome, Italy (2012)
7. Glöckner, I., Pelzer, B., Dong, T.: The LogAnswer project at QA4MRE 2011. In: CLEF 2011 Working Notes. (September 2011)
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, Berlin (2006)
10. Hartrumpf, S., Helbig, H., Osswald, R.: The semantically based computer lexicon HaGenLex. Traitement automatique des langues **44**(2) (2003) 81–105