Extending a Logic-Based Question Answering System for Administrative Texts

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. LogAnswer is a question answering (QA) system for German that uses machine learning for integrating logic-based and shallow (lexical) validation features. For ResPubliQA 2009, LogAnswer was adjusted to specifics of administrative texts, as found in the JRC Acquis corpus. Moreover, support for a broader class of questions relevant to the domain was added, including questions that ask for a purpose, reason, or procedure. Results confirm the success of these measures to prepare LogAnswer for ResPubliQA, and of the general consolidation of the system. According to the *C@1/Best IR baseline* metric that tries to abstract from the language factor, LogAnswer was the third best of eleven systems participating in ResPubliQA. The system was especially successful at detecting wrong answers, with 73% correct rejections.

1 Introduction

LogAnswer is a question answering system that uses logical reasoning for validating possible answer passages and for extracting answer phrases.[3] It was first evaluated in QA@CLEF 2008 [1]. The ResPubliQA task [2] posed some new challenges for LogAnswer:

– The JRC Acquis[4] corpus contains administrative texts that are hard to parse. Since logical processing in LogAnswer requires syntactic-semantic analyses, the parser had to be adjusted to JRC Acquis, and a graceful degradation of results had to be ensured when parsing fails.
– LogAnswer had to be extended to detect the new PURPOSE, PROCEDURE and REASON questions of ResPubliQA and to find matching answers.
– While LogAnswer used to work strictly sentence-oriented, ResPubliQA now expects a whole paragraph to be found that best answers a question.

[3] Funding by the DFG (Deutsche Forschungsgemeinschaft), FU 263/12-1, HE 2847/10-1 (LogAnswer) is gratefully acknowledged. Thanks to Tim vor der Brück for his n-gram recognizer, and to Sven Hartrumpf for adapting the WOCADI parser.

[4] see `http://langtech.jrc.it/JRC-Acquis.html`

– The main evaluation metric of ResPubliQA, the c@1 score, rewards QA systems that prefer not answering over giving wrong answers. LogAnswer computes a quality score that includes a logical validation. A threshold for cutting off poor answers had to be determined that optimizes the c@1 metric.

The overall goal of our participation in QA@CLEF was that of evaluating the various improvements of the system; this includes the refinements of LogAnswer based on lessons from QA@CLEF 2008 and the extensions for ResPubliQA.

In the paper, we first introduce the LogAnswer system. Since many aspects of LogAnswer are already described elsewhere [1,4,5], we focus on novel developments added for ResPubliQA. We then detail the results of LogAnswer and show the effectiveness of the measures taken to prepare LogAnswer for ResPubliQA.

2 Preparing LogAnswer for the ResPubliQA Task

2.1 Overview of the LogAnswer System

LogAnwer uses the WOCADI parser [6] for a deep linguistic analysis of texts and questions. After retrieving 100 candidate snippets, the system tries to prove the logical question representation from that of each candidate passage to be validated and from its background knowledge [5]. For better robustness to knowledge gaps, the prover is embedded in a relaxation loop that skips non-provable literals until a proof of the reduced query succeeds. Adding 'shallow' criteria (e.g. lexical overlap) also improves robustness. A machine learning (ML) approach computes a quality score for the text passage from this data. If several passages support an answer, then their quality scores are combined into the final answer score [7].

2.2 Improvements of Document Analysis and Indexing

Optimization of the WOCADI Parser for Administrative Language. WOCADI finds a full parse for more than half of the sentences in the German Wikipedia, but this number was only 26.2% for JRC Acquis (partial parse rate: 54%). In order to find more parsing results, a technique for reconstructing German characters ä, ö, ü and ß from ae, oe etc. was added, and the case sensitivity of the parser was switched off for sentences with many fully capitalized words. Another problem are the complex references to (sections of) regulations etc. in administrative texts, e.g. *"(EWG) Nr. 1408/71 [3]"*. We trained an n-gram recognizer using the current token and up to three previous tokens. The probability that a token belongs to a complex document reference is estimated by a log-linear model based on the probabilities for unigrams, bigrams, trigrams and four-grams. The representation of the complex name is then filled into the parsing result.

The changes to WOCADI achieved a relative gain of 12.6% in the rate of full parses, and of 5.6% for partial parses. Still, only 29.5% of the sentences in JRC Acquis are assigned a full parse. So, the extension of LogAnswer by techniques for handling non-parseable sentences had to be enforced for ResPubliQA.

Interestingly, the questions in the ResPubliQA test set were much easier to parse than the texts in the corpus: For the ResPubliQA questions, WOCADI had a full parse rate of 90% and partial parse rate (with chunk parses) of 96.4%.

Indexing Sentences with a Failed Parse. In the first LogAnswer prototype, only sentences with a full parse were indexed. But JRC Acquis is hard to parse, so we have now included sentences with a failed or incomplete parse as well. Since LogAnswer also indexes the possible answer types found in the sentences [1], the existing solution for extracting answer types had to be extended to recognize expressions of these types in arbitrary sentences.

We also complemented the special treatment of regulation names described above by a method that helps for non-parseable sentences. The tokenization of the WOCADI parser was enriched by two other tokenizers: the GermanAnalyzer of Lucene, and a special tokenizer for recognizing email addresses and URLs. New tokens found by these special tokenizers were also added to the index.

Support for New Question Categories. Trigger words (and more complex patterns applied to the morpho-lexical analysis of sentences) were added for recognizing sentences that describe methods, procedures, reasons, purposes, or goals. By indexing these types, LogAnswer can focus retrieval to matching sentences.

Beyond Indexing Individual Sentences. One novel aspect of ResPubliQA was the requirement to submit answers in the form of full paragraphs. This suggests the use of paragraph retrieval or of other means for finding answers when the relevant information is scattered over several sentences. In addition to its sentence index, LogAnswer now offers a paragraph-level and a document-level index. Moreover a special treatment for anaphoric pronouns based on the CORUDIS coreference resolver [6] was added. Whenever CORUDIS finds an antecedent for a pronoun, the antecedent is used for enriching the description of the considered sentence in the index. So, if the pronoun 'es' in a sentence refers to 'Spanien' (Spain), then 'Spanien' is also added to the index.

2.3 Improvements of Question Processing

Syntactic-Semantic Parsing of the Question. The linguistic analysis of questions also profits from the adjustments of WOCADI to administrative texts. References to (sections of) legal documents in a question are treated in a way consistent with the treatment of these constructions in answer paragraphs. Similarly, the additional tokenizers used for segmenting the texts are also applied to the question in order to generate a matching retrieval query.

Refinement of Question Classification. LogAnswer uses a rule-based question classification for recognizing question categories. Its system of 165 classification rules now also covers the new categories PROCEDURE, REASON, PURPOSE of ResPubliQA. The new classification rules have two effects: (a) the question category is identified, so that retrieval can focus on suitable text passages; and

(b) expressions like ‘What is the reason’ or ‘Do you know’ can be deleted from the descriptive core of the question used for retrieval and reasoning.

Querying the Enriched Index. The retrieval step profits from all improvements described in Sect. 2.2. Since many validation features of LogAnswer are still sentence-based, the sentence-level index was queried for each question in order to fetch the logical representation of 100 candidate sentences. For experiments on the effect of paragraph-level and document-level indexing, the 200 best paragraphs and the 200 best documents for each question were also retrieved.

Features for Candidate Quality. The validation features described in [5] (e.g. lexical overlap, answer type check. . .) were used for assessing the quality of retrieved passages. The definition of these features was extended to sentences with a failed parse in order to improve validation results in this case.

Estimation of Validation Scores. One of our lessons from QA@CLEF08 was the inadequacy of the first ML approach of LogAnswer for computing quality scores for the retrieved passages. We thus developed a new solution using rank-optimizing decision trees [5]. The result was a 50% accuracy gain of LogAnswer on the QA@CLEF 2008 test set [1]. The new models were also used for ResPubliQA. The obtained scores for individual sentences are then aggregated [7].

Optimization of the c@1 Score. The threshold θ for accepting the best answer (or refusing to answer) was chosen such as to optimize the c@1 score of LogAnswer on the ResPubliQA development set: The questions were translated into German, and LogAnswer was run on the translations. $\theta = 0.08$ was then identified as the optimum threshold, achieving a c@1 score of 0.58 on the training set. Once a retrieved sentence with top rank is evaluated better than θ, the paragraph that contains the sentence becomes the final LogAnswer result for the given question.

3 Results on the ResPubliQA 2009 Test Set for German

The results of LogAnswer in ResPubliQA 2009 and the two official baseline results [2] are shown in Table 1. The *loga091dede* run was obtained from the standard configuration of LogAnswer with full logic-based processing, while *loga092dede* was generated with the prover switched off. It simulates the case that all retrieved passages have a failed parse. Considering the number of questions with a correct paragraph at top rank, the logic-based run *loga091dede* performed best, closely followed by the shallow LogAnswer run and then the baseline runs *base092dede* and *base091dede*. LogAnswer clearly outperforms both baselines with respect to the c@1 score that includes validation quality. It was also good at detecting wrong answers: The decision to reject answers with a low validation was correct in 73% of the cases.

Table 1. Results of LogAnswer in ResPubliQA. #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
loga091dede	complete system	202	0.40	0.44
loga092dede	system w/o prover	199	0.40	0.44
base091dede	official baseline 1	174	0.35	0.35
base092dede	official baseline 2	189	0.38	0.38

Table 2. Accuracy by question category

Run	DEFINITION	FACTOID	PROCEDURE	PURPOSE	REASON
	(95)	(139)	(79)	(94)	(93)
loga091dede	0.168	0.547	0.291	0.362	0.570
loga092dede	0.137	0.554	0.291	0.362	0.559

3.1 Strengths and Weaknesses of LogAnswer

A breakdown of results by question category is shown in Table 2. LogAnswer was best for FACTOID and REASON questions. PROCEDURE and DEFINITION results were poor in both LogAnswer runs.

As to definition questions, the training set for learning the validation model (using QA@CLEF 2007/2008 questions) included too few examples for successful application of our ML technique. Thus the model for factoids (also used for REASON etc.) is much better than the model for definition questions.

Another factor is the treatment of references to definitions, e.g.

> *"Permanent pasture" shall mean "permanent pasture" within the meaning of Article 2 point (2) of Commission Regulation (EC) No 795/2004.*

It was not clear to us that such definitions by reference would not be accepted. But the main problem was the form of many definitions in JRC Acquis, e.g.

> *Hop powder: the product obtained by milling the hops, containing all the natural elements thereof.*

Since the retrieval queries for definition questions were built such that only sentences known to contain a definition were returned, many definitions of interest were skipped only because this domain-specific way of expressing definitions was not known to LogAnswer. The solution is making the requirement that retrieved sentences contain a recognized definition an optional part of the retrieval query.

The PROCEDURE result reflects the difficulty of recognizing sentences describing procedures, as needed for guiding retrieval to relevant sentences.

A breakdown of FACTOID results is shown in Table 3. Questions for countries were classified LOCATION or ORG(ANIZATION), depending on the question. OTHER and OBJECT questions were lumped together since LogAnswer

Table 3. Accuracy by expected answer types for the FACTOID category

Run	COUNT	LOCATION	MEASURE	ORG	OTHER	PERSON	TIME
	(3)	(8)	(16)	(14)	(80)	(3)	(16)
loga091dede	0.33	0.75	0.56	0.71	0.51	1.00	0.44
loga092dede	0.33	1.00	0.56	0.71	0.50	1.00	0.44

does not discern these types. In both LogAnswer runs, LOCATION, ORGANIZATION and PERSON questions clearly worked well.

3.2 Effectiveness of Individual Improvements

Recognition of References to Legal Documents. The ResPubliQA test set contains 15 questions with names of legal documents that our *n*-gram recognizer should find; the recognition rate was 87%. The benefit of a found reference is that the parser has a better chance of analyzing the question, in which case the interpretation of the reference is filled into the semantic representation. Since the recognized entities are indexed, retrieval also profits in this case.

Use of Additional Tokenizers. The special treatment of references to legal documents is only effective for parseable sentences. However, some of these references are also covered by the extra tokenizers that were added to LogAnswer. On the ResPubliQA 2009 test set, this happened for 21 questions. The benefit of analyzing regulation names like *821/68* as one token is, again, the increased precision of retrieval compared to using a conjunction of two descriptors *821* and *68*.

Effectiveness of Changes to the Retrieval Module. For ResPubliQA, sentences with a failed or incomplete parse were added to the index. Considering the 202 correct answer paragraphs in the *loga091dede* run, only 49 of these answers were based on the retrieval of a sentence of the paragraph with a full parse, while 106 stem from a sentence with a chunk parse, and 47 from a sentence with a failed parse (similar for *loga092dede*). Thus, extending the index beyond sentences with a full parse was essential for the success of LogAnswer in ResPubliQA.

Success Rate of Question Classification. Results on the success rate of question classification in LogAnswer are shown in Table 4. The recognition rules apply to the parse of a question, so the results for questions with a full parse are most informative. The classification was best for DEFINITION, REASON and FACTOID questions. The low recognition rates for PURPOSE and PROCEDURE are due to missing trigger words that signal questions of these types: *'Zielvorstellung'* (objective) was not a known PURPOSE trigger, and *'Verfahren'* (process) was not listed for PROCEDURE. Compounds of triggers were also not recognized, e.g. *Arbeitsverfahren* (working procedure), or *Hauptaufgabe* (main task). The problem can be fixed by adding these trigger words, and by treating compounds of trigger words also as triggers for a question type.

Table 4. Success rate of question classification (class-all is the classification rate for arbitrary questions and class-fp the classification rate for questions with a full parse)

Category	#questions	class-all	#full parse	class-fp
DEFINITION	95	85.3%	93	87.1%
REASON	93	73.3%	82	85.4%
FACTOID	139	70.5%	117	76.9%
PURPOSE	94	67.0%	86	72.1%
PROCEDURE	79	20.3%	72	22.2%
(total)	500	65.6%	450	70.9%

Effect of Question Classification. Since ResPubliQA does not require exact answer phrases, we wondered if recognizing the question category and expected answer type is still needed. We checked *loga091dede* and found that for all question categories except DEFINITION, results were more accurate for questions classified correctly. For definition questions, however, the accuracy for the 14 misclassified questions was 0.36, while for the 81 correctly recognized questions, the accuracy was only 0.14. The two cases differ in the retrieval query: if a definition question is identified, then an obligatory condition is added to the query that cuts off all sentences not known to contain a definition. If a definition question is not recognized, this requirement is missing. This suggests that the obligatory condition should be made an optional part of the retrieval query.

Selection of Acceptance Threshold. In the ResPubliQA runs, a threshold of $\theta = 0.08$ was used for cutting off wrong answers. The optimum for *loga091dede* would have been $\theta = 0.11$ (c@1 score 0.45 instead of 0.44). For *loga092dede*, the optimum would have been $\theta = 0.09$, but it hardly changes the c@1 score. Thus, the method for finding θ (by choosing the threshold that maximizes c@1 on the development set) was effective – it resulted in close-to-optimal c@1 scores.

3.3 Experiments with Paragraph-Level and Document Indexing.

One of the features used for determining validation scores is the original retrieval score of the Lucene-based retrieval module of LogAnswer. In order to assess the potential benefit of paragraph-level and document-level indexing, we have experimented with the *irScore* feature. Consider a retrieved candidate sentence c. Then the following variants have been tried: $irScore_s(c)$ (original retrieval score on sentence level), $irScore_p(c)$ (retrieval score of the paragraph containing c), $irScore_d(c)$ (retrieval score of the document containing c), and also the following combinations: $irScore_{ps}(c) = \frac{1}{2}(irScore_p(c) + irScore_s(c))$, $irScore_{ds}(c) = \frac{1}{2}(irScore_d(c) + irScore_s(c))$, $irScore_{dp}(c) = \frac{1}{2}(irScore_d(c) + irScore_p(c))$, and finally $irScore_{dps}(c) = \frac{1}{3}(irScore_d(c) + irScore_p(c) + irScore_s(c))$. See Table 5 for results; the $irScore_s$ configuration corresponds to *loga091dede*. The document-only result ($irScore_d$) shows that either sentence-level or paragraph-level information is needed for selecting correct answers.

Table 5. Experimental Results using Paragraph-Level and Document-Level Indexing

run	#right cand.	accuracy
irScore$_{ps}$	205	0.41
irScore$_{s}$	202	0.40
irScore$_{dps}$	198	0.40
irScore$_{p}$	196	0.40

run	#right cand.	accuracy
irScore$_{dp}$	191	0.39
irScore$_{ds}$	190	0.38
irScore$_{d}$	136	0.28

4 Conclusion

We have presented the LogAnswer QA system. Results confirm that the measures taken to prepare LogAnswer for ResPubliQA were effective. Due to the low parse rate for JRC Acquis, the use of logical reasoning in the first LogAnswer run meant only a slight benefit. On the other hand, the results in the shallow-only run show that the proposed robustness enhancements work. In general, the ResPubliQA results suggest that shallow linguistic processing or even plain passage retrieval can often identify the correct answer, but this may simply reflect that many ResPubliQA questions use the exact wording found in the answer paragraph. LogAnswer is online at `www.loganswer.de`, and in this configuration, it shows the five best answers. We found that for *loga091dede*, 60% of the questions are answered by one of the top-five paragraphs shown to the user. This number will further improve once the identified bugs are fixed, and we then expect LogAnswer to become a very useful tool for searching information in administrative texts.

References

1. Glöckner, I., Pelzer, B.: Combining logic and machine learning for answering questions. [3] 401–408
2. Peñas, A., Forner, P., Sutcliffe, R., Rodrigo, A., Forascu, C., Alegria, I., Giampiccolo, D., Moreau, N., Osenova, P.: Overview of ResPubliQA 2009: Question Answering Evaluation over European Legislation. *this volume*
3. 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, Heidelberg, Springer (2009)
4. Furbach, U., Glöckner, I., Helbig, H., Pelzer, B.: LogAnswer - A Deduction-Based Question Answering System. In: Automated Reasoning (IJCAR 2008). Lecture Notes in Computer Science, Springer (2008) 139–146
5. Furbach, U., Glöckner, I., Pelzer, B.: An application of automated reasoning in natural language question answering. AI Communications (2010) (to appear).
6. Hartrumpf, S.: Hybrid Disambiguation in Natural Language Analysis. Der Andere Verlag, Osnabrück, Germany (2003)
7. Glöckner, I.: RAVE: A fast logic-based answer validator. [3] 468–471