The LogAnswer Project at CLEF 2009

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 system, a research prototype of a question answering (QA) system for German, participates in QA@CLEF for the second time. The ResPubliQA task was chosen for evaluating the results of the general consolidation of the system and improvements concerning robustness and processing of administrative language. LogAnswer uses a machine learning (ML) approach based on rank-optimizing decision trees for integrating logic-based and shallow (lexical) validation features. The paragraph with the highest rank is then chosen as the answer to the question. For ResPubliQA, LogAnswer was adjusted to specifics of administrative documents, as found in the JRC Acquis corpus. In order to account for the low parsing rate for administrative texts, indexing, answer type recognition, and all validation features were extended to sentences with a failed parse. Moreover, support for questions that ask for a purpose, reason, or procedure was added. Compared to the first prototype of LogAnswer that participated in QA@CLEF 2008, there were no major changes in the resources employed. We have utilized the Eurovoc thesaurus for extracting definitions of abbreviations and acronyms but this knowledge was not activated by the questions in the ResPubliQA test set. Two runs were submitted to ResPubliQA: The first run was obtained from the standard configuration of LogAnswer with full logic-based processing of results, while the second run was run with the prover switched off. It simulates the performance of the system when all retrieved passages have a failed parse. The results obtained for the two runs were almost identical. Given that our parser for German has generated a useful logical representation for less than 30% of the sentences in the JRC Acquis corpus, it is not surprising that logical processing had a minor effect. A systematic analysis of the results of LogAnswer for the different question categories revealed an unfavorable decision in the processing of definition questions that will now be fixed. Moreover, questions asking for a procedure proved difficult to answer. On the positive side, the results of LogAnswer were particularly convincing for factoid questions and for questions that ask for reasons. With an accuracy of 0.40 and c@1 score of 0.44, LogAnswer also outperformed the two official ResPubliQA baselines for German.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search Process, Selection process*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Predicate Logic, Semantic networks*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing

General Terms

Experimentation, Measurement, Verification

Keywords

Logical Question Answering, Questions beyond Factoids, Passage Reranking, Robust Inference

1 Introduction

The goal of the LogAnswer project[1] is to further research into logic-based question answering. Emphasis is placed on the problem of achieving acceptable response times in the logical QA framework, and on the problem of ensuring stable results despite the brittleness of a deep linguistic analysis and of logical reasoning. An early prototype of the LogAnswer QA system that evolved from this research took part in QA@CLEF 2008. After consolidation and improvement, LogAnswer now attends the CLEF QA systems evaluation for the second time. The ResPubliQA[2] task was chosen for evaluating LogAnswer since GikiCLEF[3] requires special geographic knowledge not available to LogAnswer while the third QA system evaluation, QAST[4] is not available for German. Therefore, only ResPubliQA provided a suitable testbed for evaluating our system. The ResPubliQA question answering task is based on the JRC Acquis[5] corpus of documents related to EU administration. As opposed to earlier QA@CLEF tasks, ResPubliQA does not require the extraction of exact answers from retrieved paragraphs. But ResPubliQA introduces new difficulties that make it a demanding task for the LogAnswer system:

- The JRC corpus is characterized by administrative language. The texts are syntactically complex and contain special structures (such as references to sections of regulations, and very long enumerations of items) that are difficult to analyze syntactically. Since logical processing of questions in LogAnswer depends crucially on the success of syntactic-semantic analysis, it was important to adjust the parser to the documents found in the JRC Acquis corpus. Moreover, LogAnswer had to be equipped with fallback methods in order to ensure a graceful degradation of results if linguistic analysis or the logic-based processing of the question fails.

- Compared to earlier QA@CLEF tasks, ResPubliQA also brings significant changes with respect to the considered types of questions. There are three new question categories (PURPOSE, PROCEDURE, REASON). Moreover, even for the familiar FACTOID category, there is a shift from simple questions asking for entities with a known type (like PERSON, LOCATION) to more general questions (for example, questions asking for preconditions) with answer type OTHER in earlier QA@CLEF terminology. The LogAnswer system had to be extended to recognize these types of questions in question classification and to find suitable paragraphs in the texts.

- ResPubliQA now expects a whole paragraph to be returned as the answer to a question. While answer selection in LogAnswer used to be strictly sentence-oriented, the emphasis of ResPubliQA on answer paragraphs made it necessary to consider information from several sentences in a paragraph.

- ResPubliQA introduces the c@1 score as the primary evaluation criterion. Apart from the number of correct results found by the system, the c@1 score also takes the quality of validation into account. The QA system must thus be able to recognize bad answers,

[1]Funding of this work by the DFG (Deutsche Forschungsgemeinschaft) under contracts FU 263/12-1 and HE 2847/10-1 (LogAnswer) is gratefully acknowledged.

[2]`http://celct.isti.cnr.it/ResPubliQA/`

[3]Cross-language Geographic Information Retrieval from Wikipedia, see `http://www.linguateca.pt/GikiCLEF/`

[4]QA on Speech Transcripts, see `http://www.lsi.upc.edu/~qast/2009/`

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

and if in doubt, prefer to show no response. To this end, LogAnswer computes a quality score based on a large number of features (including logical validation). For ResPubliQA, a suitable threshold had to be found such that cutting off answers with a quality score below the threshold results in an increase of the c@1 metric.

Apart from the general consolidation of the prototype, LogAnswer was extended for ResPubliQA in order to meet these requirements. However, any customization to idiosyncratic aspects of the JRC corpus (like the special style of expressing definitions found in regulations) was deliberately avoided, in favor of developing generic methods that are useful for other corpora as well. The overall goal of our participation in QA@CLEF was that of evaluating the success of the measures taken to improve the LogAnswer system; this includes the refinements of LogAnswer based on our experience from QA@CLEF 2008, and also the extensions specifically developed for ResPubliQA (such as supporting REASON or PURPOSE questions) that enhance the question answering capabilities of the prototype.

In the paper, we first introduce the LogAnswer system. Since the architecture of LogAnswer and many details of specific solutions are already published elsewhere [1, 2, 3, 5, 6], we focus on a description of the improvements and extensions that were made for ResPubliQA. We then discuss the results obtained by LogAnswer, including a detailed analysis of the strengths and weaknesses of the system and of typical problems that were encountered. In particular, we assess the effectiveness of the measures taken to prepare LogAnswer for the ResPubliQA task.

2 System Description

2.1 Overview of the LogAnswer System

LogAnswer is a question answering system that uses logical reasoning for validating possible answer passages and for identifying the actual answer in these passages. To this end, the documents in the corpus are translated into logical form. The system then tries to prove the logical representation of the question from the logical representation of the answer passage to be validated and from its general background knowledge. In order to gain robustness against gaps in the background knowledge and other sources of errors, the prover is embedded in a relaxation loop that gradually skips non-provable literals until a proof of the simplified fragment of the query succeeds. If the validation score of the checked text passage is changed accordingly, this mechanism can help to achieve a graceful degradation of result quality in the case of errors. In addition to using a relaxation loop, the logical validation is complemented with so-called 'shallow' linguistic criteria (like the degree of lexical overlap between question and answer passage) in order to gain more robustness. A machine learning (ML) approach integrates the resulting criteria for result quality and generates a local score that judges the quality of the considered support passage (and possibly also the quality of the extracted exact answer string if such extraction takes place). In the event that a given answer is supported by several support passages, the evidence provided by the diverse support passages is aggregated in order to improve the ranking of the answer. This aggregation mechanism, which proved effective in [4], was also used in ResPubliQA because we wanted the system to benefit from aggregation. Since ResPubliQA only requires answer paragraphs but no further answer extraction, extracted precise answer strings were dropped after aggregation, and only the best validated paragraph for each question (according to the aggregated evidence) was included into the ResPubliQA result.

Despite its use of logical knowledge processing, the response times of LogAnswer are in the order of a few seconds. This is possible because the time-consuming linguistic analysis of all documents is performed prior to indexing. Therefore the retrieval module can immediately provide all retrieved answer passages together with their pre-computed semantic analysis (in the form of a MultiNet [8]). Before logical processing of any retrieved passage starts, LogAnswer computes a set of shallow linguistic features that can be assessed very quickly. These shallow features are utilized by an ML approach for a first ranking of the retrieved passages. Depending on the specified time limit for answering the question, only the best passages according to this ranking are subjected to further

Table 1: Parsing rate of WOCADI for JRC-Corpus before and after robustness enhancements and adjustment to administrative German. Results for the German CLEF news corpus and Wikipedia are shown for comparison. The 'partial parse' column includes both full and chunk parses

Corpus	full parse	partial parse
JRC-Acquis (original parser)	26.2%	54.0%
JRC-Acquis (adjusted parser)	29.5%	57.0%
CLEF News	51.8%	84.1%
Wikipedia (Nov. 2006)	50.2 %	78.6%

logical processing and validation. Exact answer strings are extracted from the variable bindings determined by proving the question representation from the representation of the support passage. Therefore every extracted precise answer is already logically validated. Compared to the usual generate-and-test paradigm of extracting a large number of potential answer strings that are then validated, this approach means a great efficiency advantage.

The general architecture of LogAnswer was presented in [1]. Some experiments concerning robustness are described in [3, 5]. Details on the E-KRHyper prover used by LogAnswer can be found in [9]. The current state of the system, including optimizations of the prover, the current set of features used for ranking candidates, and the ML technique used for learning the ranking, is described in [2].

2.2 Improvements of Document Analysis and Indexing

In the following, we describe the improvements and extension of the LogAnswer prototype that were added for ResPubliQA 2009. We begin by explaining changes related to document analysis and indexing. Following that, we detail the changes that affect question processing.

Optimization of the WOCADI Parser for Administrative Language LogAnwer uses the WOCADI parser [7] for a deep syntactic-semantic analysis of documents and questions. As shown by Table 1, the administrative language found in the JRC Acquis corpus poses severe problems to the parser: While more than half of the sentences in the German Wikipedia and in the German news corpora of the earlier CLEF evaluations are assigned a full parse, this number decreases to 26.2% for JRC Acquis. A similar picture arises if we consider all sentences that have at least a partial parse (see 'partial parse' column in Table 1). Several steps were taken in order to increase the parsing rate (and thus obtain logical representations for more sentences). First of all, an umlaut reconstruction technique was added – we noticed that German umlaut characters ä, ö, ü and also the character ß were often expanded into ae, oe, ue and ss in the texts, which resulted in parsing errors. Another observation was that many sentences in the corpus are fully capitalized. In order to improve the parsing rate for these sentences, the case sensitivity of the parser was switched off for sentences where the proportion of fully capitalized words exceeds a certain threshold.

Another problem are the complex names of regulations, resolutions etc. that abound in administrative texts. An example of such a regulation name is *"(EWG) Nr. 1408/71 [3]"*. References to such legal documents and specific sections thereof (e.g. paragraphs, articles) are highly domain specific and difficult to analyze for a general-purpose parser. In order to improve the quality of parsing results, the texts from the JRC corpus were subjected to a preprocessing step that recognizes complex names of legal documents.[6] To this end, we employed an n-gram-model considering the current token and up to three previous tokens. To account for data sparsity, the total probability of a token belonging to a section or not is estimated by a log-linear model summing

[6]Many thanks to Tim vor der Brück for developing and training the recognizer for legal document names, and to Sven Hartrumpf for adjustments and extension of the WOCADI parser.

up the logarithmic probabilities for unigrams, bigrams, trigrams and four-grams. The semantic representation of the complex name is then filled into the parsing result.

As shown by the 'adjusted parser' row in Table 1, the various changes and adjustments of the WOCADI parser achieved a relative gain of 12.6% in the rate of full parses, and of 5.6% for partial parses. Even with these improvements, only 29.5% of the sentence in the JRC Acquis corpus are assigned a full parse (and thus a useful logical representation for the prover of LogAnswer to operate on). This made it very clear that the extension of LogAnswer by techniques for handling non-parseable sentences had to be enforced for ResPubliQA.

Indexing Sentences with a Failed Parse In the first LogAnswer prototype, only sentences with a full parse were indexed. Clearly such an approach is not possible for JRC Acquis since too much content would be lost. We therefore allowed sentences with a failed or poor parse to be indexed as well. This was a non-trivial task since LogAnswer not only indexes the lexical concepts that occur in a sentence. As described in [6], the system also indexes the possible answer types found in the sentences. Since the existing solution for extracting answer types was specialized on sentences with a full parse, it had to be complemented with fallback methods that can recognize expressions of the interesting types in arbitrary sentences. Based on trigger words, regular expressions, and a custom LALR grammar for recognizing numeric expressions, temporal expressions, and measurements, the system can now reliably judge if a sentence from one of the documents contains one of the answer types of interest and index the sentence accordingly. A parse of the sentence is no longer required for answer type extraction.

We also tried to complement the special treatment of regulation names described above by a method that helps for non-parseable sentences. To this end, the tokenization computed by the WOCADI parser was enriched by the results of two additional tokenizers: the GermanAnalyzer of Lucene, and a special tokenizer for recognizing email addresses and URLs. In those cases where a token found by these special tokenizers is not contained in a token found by WOCADI, it was additionally used for indexing.

Support for New Question Categories Support for the new question types PROCEDURE, PURPOSE, and REASON has also been added to LogAnswer. For that purpose, trigger words (and sometimes more complex patterns applied to the morpho-lexical analysis of the sentences) were formulated. They are used for recognizing sentences that describe methods, procedures, reasons, purposes, or goals. If the presence of one of the new answer types is detected in a sentence, then the answer type is also indexed for that sentence. Based on the answer type indexing, LogAnswer can systematically retrieve sentences of the intended type, which helps focusing retrieval on the most promising sentences.

Apart from supporting the new question categories, the treatment of questions of the familiar types has also been improved. For example, we have experimented with the use of the Eurovoc[7] thesaurus, by indexing all sentences that contain a known abbreviation from Eurovoc and its definition with a special ABBREV marker. Including this kind of knowledge had no effect on the ResPubliQA results, however, since there was no question involving an abbreviation from Eurovoc in the test set.

Beyond Indexing Individual Sentences One novel aspect of ResPubliQA was the requirement to submit answers in the form of full paragraphs. This suggests using retrieval on the paragraph level, or at least including some information beyond the sentence level so that questions can still be answered when the relevant information is scattered over several sentences. While LogAnswer was originally based on sentence-level indexing, we have now added an alternative paragraph-level index and also a document-level index. Moreover a special treatment for anaphoric pronouns has been implemented. The WOCADI parser used by LogAnswer also includes a coreference resolver (CORUDIS, see [7]). Whenever CORUDIS establishes an antecedent for a pronoun, the description of the antecedent is used for enriching the description of the considered

[7] `http://europa.eu/eurovoc/`

sentence in the index. For example, if the sentence to be indexed contains an occurrence of the pronoun 'sie' that refers to 'Bundesrepublik Deutschland', then 'Bundesrepublik' and 'Deutschland' are also added to the index. Moreover the sentence is tagged as containing an expression that corresponds to the name of a country.

2.3 Improvements of Question Processing

In the following, we describe the changes to LogAnswer that affect the processing of a given question.

Improved Syntactic-Semantic Parsing of the Question The linguistic analysis of the question obviously profits from the adjustments of the WOCADI parser to administrative texts as well. In particular, references to (sections of) legal documents in a question are treated in a way consistent with the treatment of these constructions in the corresponding 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 The question classification of LogAnswer was extended to recognize the new question categories PROCEDURE, REASON, PURPOSE introduced by RespubliQA. Rules that cover some special cases of factoid questions (e.g. questions asking for a theme/topic and questions asking for preconditions/modalities) were also added. Moreover, the improvement of the question classification involved the inclusion of new rules for existing question types. For example, LogAnswer now supports additional ways of expressing definition questions. Overall, the number of classification rules increased from 127 to 165. The refinement of the question classification rules was based on a total of 1285 test cases, including translations of all questions from the ResPubliQA 2009 development set.

Note that there was no time for adapting the background knowledge of LogAnswer to the new question types (for example by adding logical rules that link various ways of expressing reasons or purposes). Thus the only effect of the new classification rules is the recognition of the expected answer type, and the possible elimination of expressions like *'Warum'* (why) or *'Was ist der Grund'* (What is the reason) that do not contribute anything to the meaning of the question beyond specifying the question category. The resulting core query and the expected answer type then form the basis for retrieving potential answer paragraphs.

Querying the Enriched Index The retrieval step profits from all improvements described in the subsection on document analysis and indexing. Since many of the validation features used by 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.[8] 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.

Changes in the Computed Features In the experiments, the validation features already described in [2] were used, with some refinements concerning the way in which the features are computed. In particular, the descriptors and the found answer types provided by the coreference resolution of pronouns are now included in features that depend on the matching of descriptors or of the expected vs. found answer types. Moreover the features have been generalized to retrieved sentences with a failed parse.

Improved Estimation of Validation Scores One of the main lessons from QA@CLEF08 concerning the first LogAnswer prototype was the inadequacy of the earlier ML approach for determining validation scores. After analysing the problem, we came up with a new solution

[8] LogAnswer is normally configured to retrieve 200 candidate sentences but in the ResPubliQA runs, only 100 were retrieved by mistake.

Table 2: Results of LogAnswer in ResPubliQA. Note that #right cand. is the number of correct paragraphs on top-1 position before applying the acceptance threshold and accuracy = #right cand./#questions

run	#right cand.	accuracy	c@1 score
loga091dede	202	0.40	0.44
loga092dede	199	0.40	0.44
base091dede	174	0.35	0.35
base092dede	189	0.38	0.38

based on rank-optimizing decision trees; see [2] for a description of the new method and some experimental results. As observed in [6], switching from the earlier ML approach to the new models yielded a 50% gain in the accuracy of LogAnswer on the QA@CLEF 2008 test set for German. The same models based on kMRR-optimizing decision trees for $k = 3$ were also used for generating the ResPubliQA runs of LogAnswer.[9] The resulting evaluation scores based on the evidence from individual sentences are then aggregated as described in [4].

Optimization of the c@1 Score The main evaluation metric of ResPubliQA, i.e. the c@1 score[10], rewards QA systems that validate their answers and prefer not answering over presenting a wrong answer. In order to push the c@1 score of LogAnswer, a threshold was applied to the validation score of the best answer paragraph. The idea is that results with a low validation score should rather 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 for accepting the best answer, or refusing to answer if the aggregated score falls below the threshold, was chosen such as to optimize the c@1 score of LogAnswer on the ResPubliQA development set. To this end, the ResPubliQA 2009 development set was translated into German, and LogAnswer was run on the translated questions. The subsequent determination of the optimum threshold resulted in $\theta = 0.08$ to be chosen, achieving a c@1 score of 0.58 on the training set.[11] Once a retrieved sentence with top rank is evaluated better than the acceptance threshold, the corresponding paragraph that contains the sentence is determined and returned as the final result of LogAnswer for the question of interest.

Adjustments of Resources and Background knowledge Compared to QA@CLEF 2008, there were few changes to the background knowledge of LogAnswer (see [2, 6]). Only 150 new synonyms were added. Apart from that, the logical rules and lexical-semantic relations that form the background knowledge of LogAnswer were kept stable. We have formalized a system of logical rules for treating idiomatic expressions and support verb constructions, but this extension was not yet integrated at the time of the ResPubliQA evaluation.

3 Results on the ResPubliQA 2009 Test Set for German

The results of LogAnswer in ResPubliQA 2009 and results of the two official baseline runs are shown in Table 2. The first run, *loga091dede*, used the standard configuration of LogAnswer as described in the previous section, including the use of the logic prover for computing logic-based features. In the second run, *loga092dede*, the prover was deliberately switched off and only the

[9] Note that these models were obtained from a training set with annotations for LogAnswer results for the QA@CLEF 2007 and 2008 questions. We did not try and learn special models for ResPubliQA based on the ResPubliQA development set since annotating results from the JRC Acquis corpus seemed too difficult and tedious for a non-expert of EU administration.

[10] see official ResPubliQA guidelines at `http://celct.isti.cnr.it/ResPubliQA/resources/guideLinesDoc/ResPubliQA_09_Final_Track_Guidelines_UPDATED-20-05.pdf`

[11] This result cannot be directly projected to the ResPubliQA test set, since the development set formed the basis for refining the question classification.

Table 3: Accuracy by question category

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

'shallow' features that do not depend on the results of logical processing were used for validation. The second run thus demonstrates the fallback performance of LogAnswer when no logic-based processing is possible. Both runs are based on the same results of the retrieval module using the sentence-level index. Considering the number of questions with a correct candidate paragraph on top position, the logic-based run *loga091dede* performed best, followed by the shallow LogAnswer run and then the baseline runs *base092dede* and *base091dede*. According to a quick comparison of the runs using McNemar's test, both LogAnswer runs are significantly better than *base091dede* with respect to #right cand. ($p < 0.05$), while the difference with respect to *base092dede* is not significant. On the other hand, LogAnswer clearly outperforms both baselines with respect to the c@1 score of ResPubliQA that also takes validation quality into account.

4 Error Analysis and Discussion

4.1 Strengths and Weaknesses of LogAnswer

In order to get a general impression of the strong and weak points of LogAnswer, we have prepared a breakdown of results by question category. As shown in Table 3, LogAnswer performed particularly well for FACTOID and REASON questions, with results clearly better than the average accuracy of 0.40 of both runs. The new type of PURPOSE questions performed only slighlty worse than average. However, for PROCEDURE and DEFINITION questions the results are not satisfactory.

There are several reasons for the disappointing results of LogAnswer for definition questions. First of all, LogAnswer is known to perform better for factoid questions anyway. This is because the training set used for learning the validation model of LogAnswer contains annotated results of LogAnswer for the QA@CLEF 2007 and QA@CLEF 2008 questions. These question sets include too few definition questions to allow successful application of our machine learning technique. As a result, the model for factoids (that was also used for questions of the new ResPubliQA types) is much better than the validation model used for definition questions.

Another factor is the discernment between definitions proper and references to definitions. It is quite common in regulations to define a concept by reference to a certain other document where the relevant definition can be found. An example is

> *"Dauergrünland": "Dauergrünland" im Sinne von Artikel 2 Absatz 2 der Verordnung (EG) Nr. 795/2004 der Kommission.* ("Permanent pasture" shall mean "permanent pasture" within the meaning of Article 2 point (2) of Commission Regulation (EC) No 795/2004)

Since in regulations, ordinary definitions and definitions by reference serve the same purpose, it was not clear to us that definitions by reference would not be accepted as answers to a definition question. LogAnswer did not filter out such references to definitions which resulted in several wrong answers.

The most important cause of failure with respect to definition questions, however, was the way in which definitions are expressed in the JRC Corpus. A typical definition in a regulation looks like this:

Table 4: Accuracy by expected answer types for the FACTOID category

Run	COUNT (3)	LOCATION (8)	MEASURE (16)	ORG (14)	OTHER (80)	PERSON (3)	TIME (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

> *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)

This domain-specific style of expressing definitions was not systematically recognized by LogAnswer. This had catastrophic consequences with respect to the results of LogAnswer for definition questions because the retrieval queries for definition questions are expressed in such a way that only sentences containing a recognized definition are returned. Therefore many of the definitions of interest were totally skipped simply because this particular way of defining a concept was not recognized as expressing a definition. The obvious solution is making the requirement that retrieved sentences contain contain a recognized definition an optional rather than obligatory part of the retrieval query. In addition, more ways of expressing definitions should be recognized.

The poor performance of LogAnswer for PROCEDURE questions reflects the difficulty of recognizing sentences that express procedures in the documents, as needed for guiding retrieval to the relevant sentences. Compared to the recognition of sentences that express reasons or purposes, we found this task much harder for procedures. It also happened several times that LogAnswer returned a reference to a procedure instead of the description of the procedure itself as an answer. Since we did no anticipate that this kind of result would be judged incorrect, we did not add a filter that eliminates such answers by reference.

A breakdown of the results for FACTOID questions by their expected answer type is shown in Table 4. Questions for country names were either classified LOCATION or ORG(ANIZATION) depending on the question. Questions of the OTHER and OBJECT types were lumped together since LogAnswer does not internally distinguish these types. Due to the small numbers of questions for some of the answer types, it is hard to interpret the results, but it appears that LogAnswer worked especially well for ORGANIZATION and LOCATION questions.

4.2 Effectiveness of Individual Improvements

Success of Linguistic Analysis We have already shown in Table 1 how the improvements of WOCADI have affected the parse rate for documents in the JRC corpus. But the number of sentences in the corpus with a full parse (or even a partial parse) is still low, and this has motivated our focus on developing fallback solutions for LogAnswer that will also work for non-parseable sentences. Fortunately, the questions in the ResPubliQA test set for German were much easier to parse than the administrative documents in the JRC corpus: The WOCADI parser was able to generate a full parse for 450 questions, and a chunk parse for 32 questions, so the full parse rate was 90% and the partial parse rate (including chunk parses) was 96.4%. Thus, the success rate of linguistic analysis for the questions in the ResPubliQA test set was very high. This is important since the question classification depends on the availability of a parse of the question. Note that 17 questions in the test set contain typographic or grammatical errors. The full parse rate for these ill-formed questions was only 48% and the partial parse rate was 65%. This demonstrates a clear negative effect of these ill-formed sentences on the success of parsing.

Recognition of References to Legal Documents The ResPubliQA test set contained 15 questions with references to legal documents that should be found by our n-gram based recognizer for such references. In fact, 13 of these expressions were correctly identified, while two expressions

were not recognized due to gaps in the training data. The ResPubliQA test set further contained four questions with sloppy, abbreviated references to regulations, e.g. question 146, *'Warum sollte 821/68 aufgenommen werden?'* (Why should 821/68 be adopted?) Obviously the interpretation of *821/68* as a reference to a regulation is highly domain specific. Since LogAnswer is supposed to work in arbitrary domains, it cannot be expected to treat this case correctly. However, apart from such abbreviated references that demand a special solution limited to JRC Acquis, the recognition rate of LogAnswer for references to legal documents was satisfactory. The positive effect of a correctly recognized reference is that the parser has a better chance of analyzing the question, in which case the proper interpretation of the reference to the document is inserted into the generated semantic representation. Moreover, since the recognized document names are indexed, retrieval will be guided to the proper results when the complex name is recognized as a single token.

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 additional tokenizers that have been integrated into LogAnswer. For example, the GermanAnalyzer of Lucene that serves as one of the auxiliary tokenizers correctly analyzes *821/68* as consisting of a single token. When applied to the questions in the ResPubliQA 2009 test set, these tokenizers contributed tokens not found by WOCADI for 21 questions. Specifically, the auxiliary tokenizers produced useful tokens for all questions involving references to legal documents, including the four questions that contain abbreviated references to regulations. The benefit of analyzing *821/68* as a single token is, again, the increased precision of the retrieval step compared to using a conjunction of the two descriptors *821* and *68* in the retrieval query.

Effectiveness of Changes to the Retrieval Module The most substantial change to the retrieval subsystem of LogAnswer that we introduced for ResPubliQA was the inclusion of sentences with a failed or poor parse into the index. Considering the 202 correct top-ranked paragraphs that were found in the *loga091dede* run, we notice that only 49 of these answers was based on the retrieval of a sentence of the paragraph with a full parse, while 106 correct answers were based on a retrieved sentence with a chunk parse (incomplete parse), and 47 correct answers were based on a retrieved sentence with a failed parse. A similar picture arises for *loga092dede* where 56 correct answers were based on a retrieved sentence with a full parse, 100 answers based on a sentence with a chunk parse, and 43 correct answers were based on the retrieval of a sentence with a failed parse. This clearly demonstrates that extending the index beyond sentences with a full parse was essential for the success of LogAnswer in the ResPubliQA task.

When we checked the baseline results and Gold standard results[12] for German, we noticed that the subset of JRC Acquis that we used for generating the LogAnswer runs differs from the JRC Acquis subset that can now be downloaded from the ResPubliQA web page, most likely due to a version change that escaped our attention. As a result, 74 documents of the current subset are missing in the index of LogAnswer. This difference in the considered subset of JRC Acquis resulted in the loss of up to four possible correct answers which are present in the Gold standard or the baseline runs but not represented in the index of LogAnswer.

Success Rate of Question Classification The question classification plays an important part in LogAnswer: it not only decides which phrases in a retrieved snippet can potentially answer the question, but also affects the retrieval process since possible matches with the question categories and expected answer types are also indexed. In order to assess the reliability of the classification rules of LogAnswer and their coverage of the new question categories, we have determined the success rate of the question classification of LogAnswer, as shown in Table 5. Note that the correctness of the recognized question category and (for factoid questions) also the correct recognition of the expected answer type was checked. Results for the subset of questions of a given category that have a full parse are also shown. These results are especially instructive since the

[12] see `http://celct.isti.cnr.it/ResPubliQA/index.php?page=Pages/downloads.php`

Table 5: 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%

classification rules operate on the parse of a question. Therefore the rules should work reliably on questions with a full parse (but not necessarily for the remaining questions).

The table shows that the question classification works as expected for DEFINITION questions, REASON questions and FACTOIDS. While LogAnswer achieved an acceptable (but average) recognition rate for PURPOSE questions, the recognition rate for PROCEDURE questions was very low. These findings for PURPOSE and PROCEDURE questions can be attributed to a few missing trigger words that control the recognition of these types. For example, *'Zielvorstellung'* (objective) was not included in the list of PURPOSE triggers and *'Verfahren'* (process) was not included in the list of PROCEDURE triggers. Another problem were nominal compounds of trigger words, such as *Arbeitsverfahren* (working procedure) or *Hauptaufgabe* (main task). Both problems are easy to fix. It is sufficient to add the few missing trigger words, and to allow nominal compounds that modify a known trigger word as additional trigger words for the corresponding question category.

Effect of Correct Question Classification on Results Since ResPubliQA does not require exact answer phrases to be extracted, one may ask if the recognition of question categories and the identification of the expected answer type are still essential for finding correct answers. We have checked this dependency for the *loga091dede* run. We found that for all question categories except for definition questions, the accuracy of results was better for questions that were classified correctly. However, the observed difference between the accuracy for correctly classified and misclassified questions of a given category never exceeded 6%. A surprising result was obtained for definition questions where the accuracy for the 14 misclassified questions was 0.36, while for the 81 definition questions that were correctly classified, the accuracy was only 0.14. This once again points to a problem in the processing of definition questions. The main difference in the treatment of both cases is the form of the retrieval query: if the question is recognized as a definition question, then an obligatory condition is added to the retrieval query that cuts off all sentences except those known to contain a definition. On the other hand, if a definition question is not recognized as such, then this obligatory requirement is skipped. This suggests that the obligatory condition should be dropped, or turned into an optional part of the retrieval query. Further experiments are needed in order to determine the most suitable approach.

Selection of Acceptance Threshold When generating the runs for ResPubliQA, a threshold of $\theta = 0.08$ was used for cutting off poor answers with a low validation score. In retrospect, we can say that the optimum threshold for *loga091dede* would have been $\theta = 0.11$, resulting in a c@1 score of 0.45 instead of 0.44. For *loga092dede*, the optimum threshold would have been $\theta = 0.09$. This threshold also yields a c@1 score of 0.44 after rounding to two significant digits. These findings confirm that the method for determining thresholds for accepted results (by choosing the threshold that maximizes the c@1 score on the development set) was effective. The threshold $\theta = 0.08$ determined in this way was close to the best choices, and it resulted in c@1 scores that were very close to the theoretical optima.

Table 6: 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
$irScore_{dp}$	191	0.39
$irScore_{ds}$	190	0.38
$irScore_{d}$	136	0.28

4.3 Experiments with Paragraph-Level and Document-Level Indexing

One of the features used for determining the validation score of a retrieved sentence 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 prepared additional experiments based on different choices for the corresponding *irScore* feature. Suppose that c is a retrieved candidate sentence. Then the following variants have been tried: $irScore_s(c)$ (the original retrieval score on the sentence level), $irScore_p(c)$ (the retrieval score of the paragraph that contains sentence c), $irScore_d(c)$ (the retrieval score of the document that contains c), and also the following combinations based on the arithmetic mean: $irScore_{ps}(c) = \frac{1}{2} irScore_p(c) + \frac{1}{2} irScore_s(c)$, $irScore_{ds}(c) = \frac{1}{2} irScore_d(c) + \frac{1}{2} irScore_s(c)$, $irScore_{dp}(c) = \frac{1}{2} irScore_d(c) + \frac{1}{2} irScore_p(c)$, and finally $irScore_{dps}(c) = \frac{1}{3}(irScore_d(c) + irScore_p(c) + irScore_s(c))$. The corresponding results of LogAnswer are shown in Table 6; note that the $irScore_s$ configuration corresponds to *loga091dede*. As witnessed by the poor results for $irScore_d$ compared to the other configurations, the system obviously needs either sentence-level or paragraph-level information in order to be able to select correct answer paragraphs (this was not clear in advance because LogAnswer also uses other sentence-level features). The results for the remaining configurations are very similar and do not justify a clear preference for a specific choice. In order to better exploit the information available on the paragraph and document level, we will therefore experiment with further changes to LogAnswer. This will involve the incorporation of intersentential information in other validation features, and a retraining of the validation model for the resulting system configurations.

5 Conclusion

The paper has described the current setup of the LogAnswer QA system and the changes that were made for ResPubliQA 2009. A detailed analysis of the results of LogAnswer in the ResPubliQA task has shown that most improvements were effective, but it has also revealed a problem in the treatment of definition questions and gaps in the classification rules for PURPOSE and PROCEDURE questions that will now be fixed. With its accuracy of 0.40 and c@1 metric of 0.44, LogAnswer scored better than the official baseline runs of ResPubliQA for German.

The LogAnswer prototype is also available online[13] and in actual use, the system generally presents the five top-ranked results for the given question instead of a single result. In order to assess to usefulness of LogAnswer on the ResPubliQA test set under these more realistic conditions, we have annotated the five top ranked paragraphs for each question. We then determined the MRR (mean reciprocal rank), cutting off after the first five answers, and the number of questions for which the system presents at least one correct result in the top-five list of answers shown to the user. For *loga091dede* an MRR of 0.48 was obtained.[14] Moreover, 60% of the questions are answered by one of the paragraphs in the top-five list. If we ignore the definition questions that

[13] see `http://www.loganswer.de/`, with German Wikipedia as the corpus

[14] Results for *loga092dede* are very similar.

were not adequately handled by LogAnswer for the moment, then the system presents at least one correct result for two out of three questions. Perhaps a tool with these characteristics will already be useful for searching information in administrative texts.

References

[1] Ulrich Furbach, Ingo Glöckner, Hermann Helbig, and Björn Pelzer. LogAnswer - A Deduction-Based Question Answering System. In *Automated Reasoning (IJCAR 2008)*, Lecture Notes in Computer Science, pages 139–146. Springer, 2008.

[2] Ulrich Furbach, Ingo Glöckner, and Björn Pelzer. An application of automated reasoning in natural language question answering. *AI Communications*, 2009. (to appear).

[3] Ingo Glöckner. Towards logic-based question answering under time constraints. In *Proc. of the 2008 IAENG Int. Conf. on Artificial Intelligence and Applications (ICAIA-08)*, pages 13–18, Hong Kong, 2008.

[4] Ingo Glöckner. University of Hagen at QA@CLEF 2008: Answer validation exercise. In Peters et al. [10].

[5] Ingo Glöckner and Björn Pelzer. Exploring robustness enhancements for logic-based passage filtering. In *Knowledge Based Intelligent Information and Engineering Systems (Proc. of KES2008, Part I)*, LNAI 5117, pages 606–614. Springer, 2008.

[6] Ingo Glöckner and Björn Pelzer. Combining logic and machine learning for answering questions. In Peters et al. [11]. (to appear).

[7] Sven Hartrumpf. *Hybrid Disambiguation in Natural Language Analysis*. Der Andere Verlag, Osnabrück, Germany, 2003.

[8] Hermann Helbig. *Knowledge Representation and the Semantics of Natural Language*. Springer, 2006.

[9] Björn Pelzer and Christoph Wernhard. System Description: E-KRHyper. In *Automated Deduction - CADE-21, Proceedings*, pages 508–513, 2007.

[10] Carol Peters, Thomas Deselaers, Nicola Ferro, Julio Gonzalo, Gareth J.F. Jones, Mikko Kurimo, Thomas Mandl, Anselmo Peñas, and Vivien Petras, editors. *Working Notes for the CLEF 2008 Workshop*, Aarhus, Denmark, September 2008.

[11] Carol Peters, Thomas Deselaers, Nicola Ferro, Julio Gonzalo, Gareth J.F. Jones, Mikko Kurimo, Thomas Mandl, Anselmo Peñas, and Vivien Petras, editors. *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, 2009. Springer. (to appear).