Dependency-Based Answer Validation for German

Svitlana Babych*, Alexander Henn#, Jan Pawellek#, and Sebastian Padó#

{pado,henn,pawellek}@cl.uni-heidelberg.de,
svitlana.babych@ims.uni-stuttgart.de

*: Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart
#: Institut für Computerlinguistik, Universität Heidelberg

Abstract. This article describes the Heidelberg contribution to the CLEF 2011 QA4MRE task for German. We focus on the objective of not using any external resources, building a system that represents questions, answers and texts as formulae in propositional logic derived from dependency structure. Background knowledge is extracted from the background corpora using several knowledge extraction strategies. We answer questions by attempting to infer answers from the test documents complemented by background knowledge, with a distance measure as fall-back. The main challenge is to specify the translation from dependency structure into a logical representation. For this step, we suggest different rule sets and evaluate various configuration parameters that tune accuracy and coverage. All of runs exceed a random baseline, but show different coverage/accuracy profiles (accuracy up to 44%, coverage up to 65%).

Keywords: QA4MRE, German, machine reading, question answering, logical inference, knowledge extraction

1 The CLEF 2011 QA4MRE task

The long-term goal of NLP is to build computer systems that can communicate with humans. Arguably, an important part of this enterprise is the development of semantic analysis components which allow systems to understand the information contained in the text and reason with it. This task is often called *machine reading* [15]. CLEF 2011 introduced a track on machine reading, phrasing it as a multiple-choice question answering task [21] under the name of QA4MRE. This represents a simplified version of full machine reading in the sense that systems do not have to generate answers, but only have to discriminate among a given set of possible answers (*answer validation* [18]). At the same time, the QA4MRE task was designed in a way that emphasized the role of developing a comprehensive understanding of a small text. Questions were drawn from three domains (AIDS, Climate Change, and Music and Society). For each domain, there was a small set of test documents supposed to contain the answer, and a very large background collection of documents gathered from the web. Questions were designed to involve considerable surface variation compared to the texts, and thus

Question	Was ist das Ziel von UNAIDS? *What is the goal of UNAIDS?*
Answer Candidate 1	ein Abkommen mit dem Gesundheitsminister zu treffen *to come to an agreement with the minister of health*
Answer Candidate 2	ein Konzert mit dem African Children's Choir zu geben *to give a concert with the African Children's Choir*
Answer Candidate 3	zu verhindern, dass HIV-positive Frauen schwanger werden *to avoid that HIV-positive women become pregnant*
Answer Candidate 4	UNAIDS' Außenwirkung zu vergrößern *to increase publicity for UNAIDS*
Answer Candidate 5 (correct)	zu vermeiden, dass HIV von Müttern auf Kinder übertragen wird *to avoid that HIV is transmitted from mothers to children*
Answer Sentence	Die Botschaft, die UNAIDS derzeit in der Welt verbreitet, ist, dass wir es schaffen wollen, die Übertragung des Virus von Mutter zu Kind bis 2015 praktisch zu eliminieren. *The message that UNAIDS is currently spreading is that we want to manage to virtually eliminate the transmission of the virus from mother to child until 2015.*

Table 1. An example question with answer candidates and the answer sentence

not to be answerable by simple lexical matching. Answer validation in this setting can be seen as a textual inference task [8] with potentially complex inference steps: An answer candidate answers a question if the statement obtainable by substituting the answer into the question can be inferred from the text.

For the first (primary) of ten QA4MRE runs, the use of external resources like WordNet or paraphrase resources was prohibited. This constraint emphasizes the need for deeper analysis of the texts to (a) consolidate the semantic representations of the test documents and (b) acquire additional knowledge from the background collection. Table 1 shows an example question with its answer candidates and the sentence providing the answer in the test document. The example demonstrates the properties discussed above. None of the answer candidates are contained literally in the answer. To build an inference chain from the text to the correct answer candidate, systems need to acquire the knowledge that "having a goal" has something to do with "wanting to", that "avoiding" something can mean to "eliminate" it; that "HIV" is a "virus", and so forth.

2 Strategy and Architecture

At the time of QA4MRE 2011, there was no full-fledged general-purpose textual entailment system for German, only for the related task of question grading [14]. We therefore approached question validation with a fairly simple system. In the spirit of [9], we build mainly on dependency-based normalized syntactic representations (*predicate-argument relations*) which abstract away from the surface structure, complemented by inference rules encoding synonymy and hyponymy knowledge acquired from the background collection. More specifically,

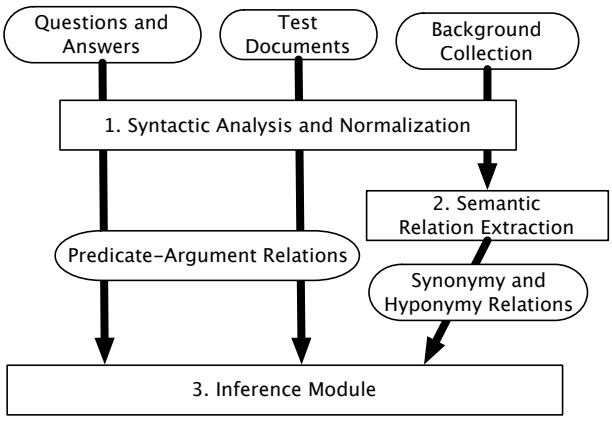

Fig. 1. Structure of the Heidelberg system for QA4MRE 2011

we assume that every relevant piece of knowledge can be expressed as a ternary relation. This includes both syntagmatic relations from actual text, e.g., dependency relations *(Peter,subj,sleeps)* and paradigmatic relations, i.e. type-level semantic relations *(natural gas,hypernym,energy source)* and semantic relatedness *(book,relatedTo,story)* acquired from the background corpus. We treat complex structures (like dependency graphs) simply as sets of such binary relations, and use the YAML file format as a universal exchange format among modules [3].

The overall structure of the system is shown in Figure 1. It consists of three modules. In the first module, all types of language input (questions, answers, test and background documents) are preprocessed, dependency-parsed, and normalized, to meet the first need outlined in the introduction (consolidated representation). The second module extracts semantic relations from the background document collection based on distributional similarity and shallow rules, thus addressing the second need from the introduction (acquisition of additional information). Finally, the third module attempts to infer each answer candidate, combined with the question, from the test document.

3 Modules in Detail

3.1 Syntactic Analysis and Normalization

This module creates dependency-syntactic structures for the German input texts (questions, answers, test and background documents). We first perform sentence splitting with the regular expression-based tokenizer by Sebastian Nagel[1] and then run the MATE dependency parsing toolkit [4] which is among the best available dependency parsers for German. After parsing, we perform a number of normalization steps whose general motivation is to bring the dependency output

[1] `http://www.cis.uni-muenchen.de/~wastl/misc/`

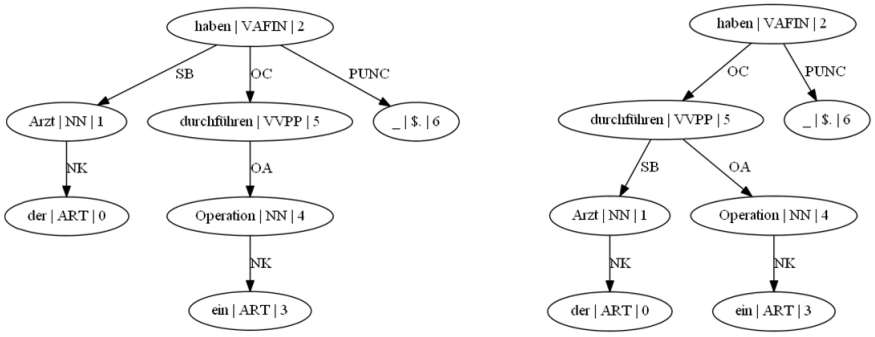

Fig. 2. German sentence in perfective tense (auxiliary + participle): "Der Arzt hat eine Operation durchgeführt *The doctor has performed an operation*". Left: parser output. Right: normalized version.

of the parser closer to our semantic intuitions, similar to [13]. Our goal is to make the representations of answers and their supporting textual evidence more similar to one another, abstracting away from surface variability. All normalization steps are realized as dependency tree transformations.

Specifically, we deal with the four most frequent phenomena that we identified. The first one is passive sentences, which we transform into an active representation, including changing the edge labels. The second is prepositional phrases, where we delete the preposition node and encode this information in the edge label. The third one is coordinations, where we expand the second conjunct which is by default only realized in a reduced form. These three phenomena can be treated in German fairly similar to English [13]. However, our final phenomenon – German verb complexes – is language-specific and warrants discussion. In English declarative sentences, verb complexes with one finite and (at least) one infinite verb – e.g., auxiliary + participle, or modal + infinitive – are, as a rule, realized contiguously. In contrast, German main clauses must realize the finite verb in second position and the infinite verb in clause-final position. The MATE parser is trained on a version of the German TIGER treebank [6] converted into dependencies. According to the TIGER annotation guidelines, sentence-initial arguments are attached to the second-position verb, while all others are attached to the clause-final verb. The left-hand side of figure 2 shows the dependency tree for "Der Arzt hat eine Operation durchgeführt", where the sentence-initial subject (doctor) is attached to the finite auxiliary and the object (operation) to the participle. Following our our semantic intuition, we move all arguments to attach to the semantic head of the sentence (cf. the tree on the right-hand side).

3.2 Knowledge Extraction from the Background Collection

We decided to concentrate on extracting two types of inference rules from the background corpus, namely hyponymy and synonymy. These two relations

Regular Expression Pattern	hypernym RE group
`(\S+)/N. wie/\S+ zum/\S+ Beispiel/\S+ (\S+)/N.`	1
`(\S+)/N. ist/\S+ eine?/\S+ (\S+)/N.`	2
`(\S+)/N. wie/\S+ etwa/\S+ (\S+)/N.`	1
`(\S+)/N. einschließlich/\S+ (\S+)/N.`	1
`(\S+)/N. und/\S+ andere/\S+ (\S+)/N.`	2
`(\S+)/N. oder/\S+ andere/\S+ (\S+)/N.`	2

Table 2. The patterns used to extract hyponymy relations. The patterns assume that each word is followed by a slash and its part of speech, such as *Energie/NN – energy/N*.

Similarity Measure	Threshold	Similarity Measure	Threshold
Cosine distance	0.8	Dice coefficient	0.7
GCM	0.7	Hindle	300.0
Jaccard	0.7	Lin	0.7
Balanced Average Precision	0.6		

Table 3. Thresholds for similarity measures.

frequently contribute to bridging the "lexical gap" between answer candidates and textual evidence, and can be acquired using well-established methods.

The first approach we followed was the extraction of hypernym-hyponym pairs with so-called *Hearst Patterns* [11]. Their adaptation to German [10] consists of six regular expression patterns (listed in table 2) which we applied to the background corpora. Our second approach to knowledge extraction was based on vector space models [20]. Given the large size of the background corpus, we constructed dependency-based vectors to take advantage of their better ability to identify close semantic similarity when sparsity is not an issue [16]. To limit computation time and memory consumption, all words occurring at least 50 times in one of our task's background corpora have been included as target words in the vector space. To extract synonyms from the spaces, we used a range of symmetrical similarity measures to compute vector similarity, including Cosine, Dice, GCM, Hindle, Jaccard, and Lin, and added all pairs above some threshold, optimized for recall, to a database of inference rules (cf. Table 3).

In contrast to synonymy, hyponymy is an asymmetrical relationship which should therefore not be amenable to extraction with symmetrical similarity measures. We therefore identified hypernymy using the asymmetrical *Balanced Average Precision* similarity measure [12]. This measure judges the relevance of the broader term's features for the narrower term (based on feature ranks) and penalizes short (e.g., vague) vectors. Again, we added pairs above a certain threshold (Table 3) to a database of inference rules.

3.3 Inference

We take the classical "logical inference" approach to deciding whether an answer follows from the text [5]. We represent the answer candidate C, the test document

T, and the background knowledge base $\mathcal{B}$ (i.e., our inference rules), as logical formulae and test the validity of the following formula:

$$(T \wedge \mathcal{B}) \vdash C \tag{1}$$

The main problem is how to represent T and C in logical terms. In an ideal world, we would obtain a complete representation of the sentence meaning as provided by a full syntax-semantics interface. However, wide-coverage translation of natural language into logics is still essentially an open research problem. Additionally, if T and A are represented by strong logical representations, then we need an equally strong background knowledge base $\mathcal{B}$ which can, for example, license paraphrastic variation between T and C (*invent* $X \rightarrow$ *be the first to think of* X). This problem has been approached e.g. by acquiring meaning postulates from WordNet [19], but in the absence of manually vetted knowledge sources, as we assume, Formula (1) will be valid for only a small fraction of all cases [5].

We address this problem by deriving weaker representations of the linguistic structures that essentially encode their dependency relations in propositional logic, experimenting with different parameters in the process. We take advantage of the fact that we have to solve only a multiple-choice task by searching for construction methods in which exactly one of the answers can be proved: if no answer can be proved, the representation is too strong; if more than one answer can be proved, it is too weak. The names of the binary parameters of this process are marked in boldface and will be used in the next chapter to explain the evaluation results.

Turning test documents into propositional formulae *T*. Our goal is to translate dependency structures into propositional logical formulae. The first, optional, step is to prune the dependency structures by removing nodes with no semantic content (option **DropPOS**). If this option is selected, we remove all leaf nodes with the parts of speech ART (article) and APPR (prepositions[2]) as well as PWAT, PWAV, PWS (question words – see below for details). The next step is to decide on the shape of the literals. We developed four different rulesets which we will demonstrate on the example of the node "durchführen/perform" from the right-hand side of Figure 2 and its two outgoing edges (SB to "Arzt/doctor" and OA to "Operation/operation").

- The **Unary** ruleset builds a literal for each dependency edge that consists only of the head and the dependent. For the above-mentioned edges, we obtain `perform(doctor)` and `perform(operation)`.
- The **LabeledUnary** ruleset builds a literal for each dependency edge like Unary, but also includes the edge label. For the example: `SB(perform,doctor)` and `OA(perform,operation)`.
- The **Binary** ruleset builds a literal for each pair of dependency edges with the same head without taking edges into account: `perform(doctor,operation)`.

[2] Note that prepositions should have been re-encoded as edge labels during the preceding syntactic normalization.

– The **Unrestricted** ruleset turns each dependency subtree of depth one (i.e., each head with all of its dependents) into one n-ary literal. For the current example, the result is identical to the output of Binary.

The logical representation of a test document is the conjunction of all its literals. Several rulesets are activated, which leads to a certain amount of redundancy.

Turning a question-answer pair into a propositional formula *C*. In QA4MRE, answers are usually not complete propositions, but only words or short phrases. Such answers must be combined with the questions in order to obtain a complete representation of the proposition conveyed by the answer candidate, such as "Is it the goal of UNAIDS to increase publicity for itself" for answer candidate 4 in Table 1. We therefore translate the linguistic realizations of question and answer candidates into propositional logic as detailed above, and combine them into a single formula in a way that is influenced by two parameters.

The first parameter is **WhSubstitution** (question word substitution) concerns the handling of single-word answers. If this option is true, then we simply replace the interrogative pronoun ("who"/"what") in the question with the answer word. If it is false, then we attempt to reconstruct a logical representation, i.e., a literal, for the answer. Recall that our literals must correspond to dependency edges. We therefore search in the dependency representation of the test document for the most frequent head which had the answer word as its argument. As an example, assume that the test document mentioned medicine being an academic discipline. If the question concerns somebody's occupation, and the answer is "medicine", we then obtain as representation for this answer candidate the literal `discipline(medicine)`, corresponding to the more detailed multi-word answer candidate "the discipline of medicine". We also construct answers by searching for heads which had hyponyms and synonyms of the answer as their argument. Additionally we also construct an answer by combining the answer word with the question's head (which in general leads to very similar results as question word substitution).

The second parameter is **ANDonly** which decides how the literals of question and answer candidate are combined into the formula C. If ANDonly is true, C is the conjunction of all literals of answers in the answer candidate, a representation that is optimized for precision. If ANDonly is false, representations for the answer and question are first formed as disjunctions over the respective literals, and then combined by conjunction. This formulae is not very precise (since it is valid as soon as one literal of question and answer each is valid), but it may be informative enough for a multiple-choice selection task (cf. the discussion above).

Proving validity and the role of background knowledge. Finally, we use the theorem prover CVC3 [2] to test the validity of the resulting formulae.[3]

[3] CVC3 is in fact a solver for predicate logic, even though currently all of our formulae are in propositional logic. We use CVC3 for future compatibility to allow, for example, the formulation of PL1 inference rules for predicates with valency mappings.

Run ID	1	2	3	4	5	6	7	8	9
Unary	T	T	T	T	T	T	T	T	T
LabeledUnary	T	T	T	F	F	F	F	F	T
Binary	T	T	T	T	T	T	T	F	T
Unrestricted	T	T	T	T	T	T	T	F	T
WhSubstitution	F	F	F	F	T	F	F	T	T
ANDonly	F	F	F	F	F	T	T	F	F
AllKnowledgeAtOnce	T	T	F	F	F	F	F	F	F
Threshold	F	F	F	F	F	F	T	F	F
DropPOS	T	T	T	T	T	T	T	F	T
RankUnprovable	F	F	F	F	F	F	F	F	T

Table 4. Submitted runs and their features

As discussed above, the basis of our approach is to attempt to prove the five formulae in parallel for a growing body of background knowledge $\mathcal{B}$, i.e., synonymy and hyponymy relations. Specifically, we divide the relations generated by the knowledge extraction component (Section 3.2) into five categories according to decreasing semantic similarity. In other words, we first attempt to prove the answer candidates when taking only fairly certain background knowledge into account, and if this is not possible, we proceed to less certain background knowledge. As soon as at least one of the answers can be proved, we stop. This process is skipped by setting the option **AllKnowledgeAtOnce**, which adds the complete body of background knowledge at once.

At the end of this process, we can have different outcomes. If none of the answers can be proved, we do not make a prediction. If exactly one answer can be proved, we return this answer. If more than one answer can be proved, we require a tie-breaker to choose among these candidates. For this end, we score each derivation based on the similarity scores of the background knowledge that was employed, effectively interpreting similarity scores as confidence values. Questions where more than one answer receives exactly the same score must however still be left unanswered. We also experimented with the option **Threshold** which introduced a fixed threshold that answer scores had to exceed.

Finally, we experimented with a fallback mode called **RankUnprovable** where answers C that could not be proved (and were thus normally discarded) were included in the output of the system. The decision between these answers was made based on a simple distance metric that quantifies the "closeness" of answer candidates to test documents.

4 Evaluation

We submitted a total of nine runs, as permitted by the QA4MRE guidelines. Preprocessing and knowledge extraction were held constant across runs; we only varied the parameters of the inference step described in Section 3.3, as shown in Table 4. We originally planned to vary only one parameter at a time to observe

Run ID	# Correct (c)	# Answered (a)	Accuracy (c/a)	Coverage (a/120)	C@1
1	9	27	0.33	0.23	0.13
2	19	62	0.30	0.52	0.23
3	18	63	0.29	0.53	0.22
4	15	57	0.26	0.48	0.18
5	13	56	0.23	0.47	0.17
6	13	33	0.39	0.28	0.19
7	11	25	**0.44**	0.21	0.16
8	14	50	0.28	0.42	0.18
9	22	82	0.27	**0.68**	**0.24**

Table 5. Results for runs (120 questions). Best result for each statistic in boldface.

the impact of parameters, but had to deviate from this idea to sample a larger part of the parameter space. Consequently, we decided to use a quantitative approach to analyze the influence of run parameter on run performance.

We analyzed our runs with a logistic regression model which have previously been used successfully to explain the influence of features in data [7]. We used the runs' properties (cf. Table 4) as predictors x, and the run performance as the response variable y to be predicted. Logistic regression models have the form

$$p(y = 1) = \frac{1}{1 + e^{-z}} \text{ with } z = \sum_i \beta_i x_i \tag{2}$$

where p is the probability of the response variable taking some value and β_i the coefficient assigned to predictor x_i. Model estimation sets the parameters β to maximize the likelihood of the data. In the current setup, we are interested in analyzing the predictor weights: for each predictor x_i, we can test the hypothesis that it significantly contributes to predicting the response.

We first attempted to fit a model to the official evaluation metric of QA4MRE 2011, namely C@1. However, we failed to find any significant predictors. While the fact that we have relatively few runs can definitely play a role, we attribute this failure to the property of C@1 to combine coverage and accuracy into a single figure of merit [17]. We found that we were more successful by investigating the influence of our predictors for coverage and accuracy separately (cf. Table 5).

4.1 Predicting coverage

Column 5 in Table 5 (Coverage) shows the coverage figures for our different runs. We see that the runs fall into three groups: relatively high coverage (run 9, 68%), medium coverage (runs 2-5 and 8, 40-55%), and low coverage (runs 1, 6, and 7, $< 30\%$). The model with which we analysed these coverage figures revealed two significant predictors with a negative influence on coverage when set to true, namely ANDonly and AllKnowledgeAtOnce. There was also one highly significant predictor with a positive influence on coverage, namely RankUnprovable.

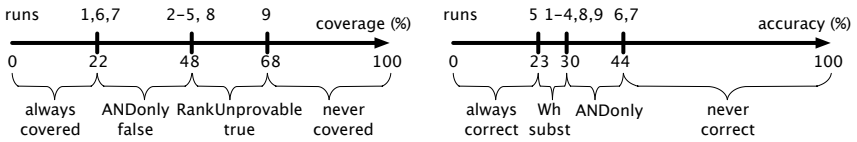

Fig. 3. Question classes according to coverage (left) and accuracy (right)

These results tie in well with our intuition about the inference approach that we use. Our model does does not make a prediction if one of two complementary situations take place: (a) none of the answers can be inferred, or (b) there are two or more answers with the same score among which we cannot decide. Since ANDonly means that the answer candidate C is a pure conjunction of the question and answer literals, setting it to true results in more cases of type (a). Conversely, RankUnprovable attempts to answer also questions where none of the answers is provable as a whole, reducing the number of (a) cases. Finally, if AllKnowledgeAtOnce is true, more ties occur, thus more cases of type (b).

To gain a better qualitative understanding of these cases, we make the simplifying, but largely warranted, assumption that our runs coincide on the questions that they can and cannot cover. This gives rise to four classes of questions with respect to coverage, shown on the left-hand side of Figure 3. Class 1 consists of questions that are always covered, even in an ANDonly setting. An example[4] is question 1-1-3, *What country is Nelson Mandela from?*.

Class 2 consists of questions where the conjunction of answer and question cannot be inferred. An example is question 2-7-10, *What could happen if the amount of CO2 in the atmosphere is not reduced?*. This question, paired with the correct answer, results in a fairly complex formula for C, with more than ten literals. If not all literals are connected by conjunctions (ANDonly), if cannot be proved, but it can if ANDonly is set to false.

Class 3 coverts cases that cannot be decided by the prover, but can be decided by the distance metric when RankUnprovable is true. An example is 2-3-5, *What could be a consequence of the reduction of arctic ice?*. We cannot prove this question together with any answer because the question contains a description of a causal relation, "a consequence of", which is not expressed in the same way in the test document. The distance metric can, however, make a prediction.

Finally, there are questions (Class 4) which none of our runs covered, such as 1-1-5, *What is Annie Lennox' profession?* The correct answer would be *musician*. However, none of the answers can be proved, which means that runs 1 to 8 abstain from answering. Run 9 attempts to apply the distance metric, but runs into a distance tie between the answers *musician* and *dancer* both of which are equally close to the test document. A further contributing factor for this question was that the parser returned an incorrect analysis of the question, which directly led to an inappropriate representation for C.

[4] For convenience, we present all examples in English. The IDs that we provide have the shape [TopicID]-[DocID]-[QuestionID].

4.2 Predicting accuracy

The fourth column in Table 5 lists accuracy figures for the different runs, where accuracy is computed as the ratio of correctly answered questions to answered questions. Again, we see three groups, although the overall accuracy is quite low, and the groups are much closer together. One run has very low accuracy (run 5 at 23%), most runs show middling accuracy (runs 1-4, 8, 9) and two runs have fair accuracy (runs 6 and 7, the best run, with 44% accuracy). In terms of accuracy, all of our runs beat a random baseline (at 20%). Due to the fact that the accuracy figures are clustered together more closely, our logistic regression model yielded only one predictor with significant (positive) impact on accuracy, namely ANDonly. As discussed above, ANDonly leads to stronger answer representations which reduce the risk for false positives. Thus, ANDonly is a true precision/recall trade-off: activating it yields higher-precision runs, deactivating it higher-recall runs. Another factor which is not significant in the regression but which we consider important to explain the performance of run 5 is WhSubstitution (question word substitution). The heuristic we employed to deal with single-word answers for interrogative pronouns (cf. Section 3.3) yielded only mixed results: too frequently, the occurrences of the answer candidate in the test document that form the basis for its interpretation were not related to the question.

In sum, we can distinguish again, though at the risk of oversimplification, four groups of questions, shown on the right-hand side of Figure 3: (1), those that are answered correctly by all runs; (2), those where question word substitution leads to wrong answers; (3), those that can be answered correctly if ANDonly is true; and (4), those that no run manages to answer correctly.

An interesting example of group (3) is question 1-3-10, *Who wrote 'People do stupid things - that's what spreads HIV'?*, with the correct answer *1: Elizabeth Pisani*. Two other answer candidates show a lot of overlap with the correct answer (*a friend of Elizabeth Pisani's* and *Elizabeth Pisani's brother*). If ANDonly is false, and consequently not the complete answer has to be proved, some of the runs mistakenly return one of the two confounds as the correct answer.

An analysis of the group (4) errors made by run 7, our highest-accuracy run, highlights the limitations of our current system architecture. One important limitation is the fact that we consider literals (i.e., individual dependency edges) largely in isolation. This makes our system vulnerable to questions where (parts of) answers are already contained in the test document. For example, one of the answer candidates for question 2-5-4, *Why are there more wildfires now?*, is *Global warming*, a term which occurs in the test document several times. Since for *why*-questions, we in effect try to prove a conjunction of question and answer, we check whether the test document contains *There are more wildfires* and *Global warming*. This is the case, and we return *Global warming* as the (incorrect) answer – the correct answer would have been *Less water*.

A second frequent problem is the inadequacy of our background knowledge extraction. For example, question 2-5-2 is *Where is the third largest ice mass in the world?*, with the correct answer candidate being *Asia*. The test document

mentions only the Himalaya (mountains). If we had had access to the meronymy (or location) relation *the Himalaya is a part of (or located in) Asia*, we would have been able to prove this answer, but we did not attempt to extract meroynym relations. Consequently, our systems attempted to prove increasingly weaker versions of the answers and ended up with incorrect answers. More generally speaking, the problem is that our current system has a very impoverished notion of background knowledge. Even within the covered relation (hyponymy), coverage is far from perfect. Outside hyponymy, a large number of relevant semantic relations are excluded both at the lexical level (e.g., meronymy or entailment relations between verbs) and at the lexico-syntactic level (paraphrases), all of which can play an important role in entailment [1].

5 Conclusions

In our contribution to the CLEF 2011 QA4MRE task, we focussed on answer validation completely without the use of any external knowledge resources, merely extracting some background knowledge from the provided background corpora. Our approach was fairly straightforward and consisted in a fairly direct translation of dependency trees into proposional logic formulae, with each edge contributing one literal. Several parameters determine the strength of the resulting formulae. We implemented a back-off proving setup where we proceeded from stronger to weaker logical representations until we were able to prove at least one of the answers. Cases of ties are resolved by a simple distance metric.

We were happy to find that our best runs showed accuracies of around 40%; however there is a clear inverse relation between coverage and accuracy, with the best runs showing the lowest coverage. This clearly shows that improvements are necessary with respect to both the precision and the recall of our approach, and our data analysis has yielded a number of clear directions for improvements. As for recall, we mainly need to improve the knowledge acquisition techniques with which we extract background knowledge from large corpora and scale them up to a larger set of semantic relations, notably paraphrase. With regard to precision, we feel that we need to improve the handling of different question types and go beyond simply matching the linguistic material in the question with the test document. In particular for causal questions, our current approach is much too restricted. More generally, we would like to generalize our current text normalization step into a more sophisticated syntax-semantics interface that maps input texts onto "more semantic" knowledge representation structures.

References

1. Bar-Haim, R., Szpektor, I., Glickman, O.: Definition and analysis of intermediate entailment levels. In: Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment. pp. 55–60. Ann Arbor, MI (2005)
2. Barrett, C., Tinelli, C.: CVC3. In: Proceedings of CAV. Lecture Notes in Computer Science, vol. 4590, pp. 298–302. Springer-Verlag (2007), berlin, Germany

3. Ben-Kiki, O., Evans, C.: Yaml ain't markup language specification, version 1.2. http://www.yaml.org/spec/1.2/spec.html (2009)
4. Bohnet, B.: Top accuracy and fast dependency parsing is not a contradiction. In: Proceedings of COLING. pp. 89–97. Beijing, China (2010)
5. Bos, J., Markert, K.: Recognising textual entailment with logical inference. In: Proceedings of EMNLP. pp. 628–635. Vancouver, BC (2005)
6. Brants, S., Dipper, S., Hansen, S., Lezius, W., Smith, G.: The TIGER treebank. In: Proceedings of the Workshop on Treebanks and Linguistic Theories. Sozopol, Bulgaria (2002)
7. Bresnan, J., Cueni, A., Nikitina, T., Baayen, H.: Predicting the dative alternation. In: Cognitive Foundations of Interpretation, pp. 69–94. Royal Netherlands Academy of Science (2007)
8. Dagan, I., Glickman, O., Magnini, B.: The PASCAL recognising textual entailment challenge. In: Machine Learning Challenges, Lecture Notes in Computer Science, vol. 3944, pp. 177–190. Springer (2006)
9. Dolan, W.B., Richardson, S.D., Vanderwende, L.: MindNet: acquiring and structuring semantic information from text. Tech. rep., Microsoft Research (1998)
10. Granitzer, M., Augustin, A., Kienreich, W., Sabol, V.: Taxonomy extraction from german encyclopedic texts. Tech. rep., Graz University of Technology (2009)
11. Hearst, M.A.: Automatic acquisition of hyponyms from large text corpora. In: Proceedings of COLING. Nantes, France (1992)
12. Kotlerman, L., Dagan, I., Szpektor, I., Zhitomirsky-Geffet, M.: Directional distributional similarity for lexical inference. Nat. Lang. Eng. 16(4), 359–389 (2010)
13. de Marneffe, M.C., Manning, C.D.: The Stanford typed dependencies representation. In: Proceedings of the COLING Workshop on Cross-Framework and Cross-Domain Parser Evaluation. pp. 1–8. Manchester, UK (2008)
14. Meurers, D., Ziai, R., Ott, N., Kopp, J.: Evaluating answers to reading comprehension questions in context: Results for German and the role of information structure. In: Proceedings of the EMNLP TextInfer 2011 Workshop on Textual Entailment. pp. 1–9. Edinburgh, Scotland, UK (2011)
15. Norvig, P.: Inference in text understanding. In: Proceedings of AAAI. pp. 561–565. Seattle, WA (1987)
16. Padó, S., Lapata, M.: Dependency-based construction of semantic space models. Computational Linguistics 33(2), 161–199 (2007)
17. Peñas, A., Rodrigo, A.: A simple measure to assess non-response. In: Proceedings of ACL/HLT. pp. 1415–1424. Portland, OR (2011)
18. Peñas, A., Rodrigo, Á., Sama, V., Verdejo, F.: Testing the reasoning for question answering validation. Journal of Logic and Computation 18, 459–474 (2008)
19. Tatu, M., Moldovan, D.: A semantic approach to recognizing textual entailment. In: Proceedings of EMNLP. pp. 371–378. Vancouver, BC (2005)
20. Turney, P.D., Pantel, P.: From frequency to meaning: Vector space models of semantics. Journal of Artificial Intelligence Research 37, 141–188 (2010)
21. Vanderwende, L.: Answering and questioning for machine reading. In: Proceedings of the AAAI Spring Symposium. Stanford, CA (2007)