Using Syntactic Information for Improving *Why*-Question Answering

Suzan Verberne, Lou Boves, Nelleke Oostdijk and Peter-Arno Coppen
Department of Linguistics
Radboud University Nijmegen
s.verberne@let.ru.nl

Abstract

In this paper, we extend an existing paragraph retrieval approach to *why*-question answering. The starting-point is a system that retrieves a relevant answer for 73% of the test questions. However, in 41% of these cases, the highest ranked relevant answer is not ranked in the top-10. We aim to improve the ranking by adding a re-ranking module. For re-ranking we consider 31 features pertaining to the syntactic structure of the question and the candidate answer. We find a significant improvement over the baseline for both success@10 and MRR@150. The most important features for re-ranking are the baseline score, the presence of cue words, the question's main verb, and the relation between question focus and document title.

1 Introduction

Recently, some research has been directed at problems involved in *why*-question answering (*why*-QA). About 5% of all questions asked to QA systems are *why*-questions (Hovy et al., 2002). They need a different approach from factoid questions, since their answers cannot be stated in a single phrase. Instead, a passage retrieval approach seems more suitable. In (Verberne et al., 2008), we proposed an approach to *why*-QA that is based on paragraph retrieval. We reported mediocre performance and suggested that adding linguistic information may improve ranking power.

In the present paper, we implement a similar paragraph retrieval approach and extend it by adding a re-ranking module based on structural linguistic information. Our aim is to find out whether syntactic knowledge is relevant for discovering relations between question and answer, and if so, which type of information is the most beneficial.

In the following sections, we first discuss related work (section 2). In sections 3 and 4, we introduce the data that we used for development purposes and the baseline retrieval and ranking method that we implemented. In section 5, we present our re-ranking method and the results obtained, followed by a discussion in section 6, and directions for further research in section 7.

2 Related work

A substantial amount of work has been done in improving QA by adding syntactic information (Tiedemann, 2005; Quarteroni et al., 2007; Higashinaka and Isozaki, 2008). All these studies show that syntactic information gives a small but significant improvement on top of the traditional bag-of-words (BOW) approaches.

The work of (Higashinaka and Isozaki, 2008) focuses on the problem of ranking candidate answer paragraphs for Japanese *why*-questions. They find a success@10 score of 70.3% with an MRR of 0.328. They conclude that their system for Japanese is the best-performing fully implemented *why*-QA system. In (Tiedemann, 2005), passage retrieval for Dutch factoid QA is enriched with syntactic information from dependency structures. The baseline approach, using only the BOW, resulted in an MRR of 0.342. With the addition of syntactic structure, MRR improved to 0.406.

The work by (Quarteroni et al., 2007) considers the problem of answering definition questions.

They use predicate-argument structures (PAS) for improved answer ranking. Their results show that PAS make a very small contribution compared to BOW only (F-scores 70.7% vs. 69.3%).

The contribution of this paper is twofold: (1) we consider the relatively new problem of *why*-QA for English and (2) we not only improve a simple passage retrieval approach by adding syntactic information but we also perform extensive feature selection in order to find out which syntactic features contribute to answer ranking and to what extent.

3 Data

As data for developing and testing our system for *why*-QA, we use the Webclopedia question set by (Hovy et al., 2002). This set contains questions that were asked to the online QA system *answers.com*. 805 of these questions are *why*-questions. As answer corpus, we use the off-line Wikipedia XML corpus, which consists of 659,388 articles (Denoyer and Gallinari, 2006). We manually inspect a sample of 400 of the Webclopedia *why*-questions. Of these, 93 have an answer in the Wikipedia corpus. Manual extraction of one relevant answer for each of these questions results in a set of 93 *why*-questions and their reference answer. We also save the title of the Wikipedia article in which each of the answers is embedded, in order to be able to evaluate document retrieval together with answer retrieval.

4 Paragraph retrieval for *why*-QA

4.1 Baseline method

We index the Wikipedia XML corpus using the Wumpus Search Engine (Buttcher, 2007). In Wumpus, queries can be formulated in the GCL format, which is especially geared to retrieving XML items. Since we consider paragraphs as retrieval units, we let the engine retrieve text fragments marked with <p> as candidate answers.

We implement a baseline method for question analysis in which first stop words are removed[1]. Also, any punctuation is removed from the question. What remains is a set of question content words. Next, we automatically create a query for each question that retrieves paragraphs containing (a subset of) these question terms. For ranking the paragraphs retrieved, we use the QAP algorithm created by MultiText, which has been implemented in Wumpus. QAP is a passage scoring algorithm specifically developed for QA tasks (Buttcher et al., 2004). For each question, we retrieve and rank the top 150 of highest scoring answer candidates.

4.2 Evaluation method

For evaluation of the results, we perform manual assessment of all answers retrieved, starting at the highest-ranked answer and ending as soon as we encounter a relevant answer[2]. Then we count the proportion of questions that have at least one relevant answer in the top n of the results for $n = 10$ and $n = 150$, giving us success@10 and success@150. For the highest ranked relevant answer per question, we determine the reciprocal rank (RR). If there is no relevant answer retrieved by the system at $n = 150$, the RR is 0. Over all questions, we calculate the Mean RR (MRR@150).

We also measure the performance of our system for document retrieval: the proportion of questions for which at least one of the answers in the top 10 comes from the reference document (success@10 for document retrieval) and the MRR@150 for the highest position of the reference document[3].

4.3 Results and discussion

Table 1: Baseline results for the *why* passage retrieval system for answer retrieval and document retrieval in terms of success@10, success@150 and MRR@150

	S@10	S@150	MRR@150
Answer retrieval	43.0%	73.1%	0.260
Document retrieval	61.8%	82.2%	0.365

There are two possible directions for improving our system: (1) by improving retrieval and (2) by improving ranking. Since success@150 is 73.1%, for 68 of the 93 questions in our set at least one relevant answer is retrieved in the top 150. For the other 25 questions, the reference answer was not included in the long list of 150 results.

In the present paper we focus on improving answer *ranking*. The results show that for 30.1% of

[1]To this end the stop word list is used that can be found at *http://marlodge.supanet.com/museum/ funcword.html*. We use all categories except the numbers and the word *why*

[2]We don't need to assess the tail since we are only interested in the highest-ranked relevant answer for calculating MRR

[3]Note that we consider as relevant all documents in which a relevant answer is embedded. So the relevant document with the highest rank is either the reference document or the document in which the relevant answer with the highest rank is embedded.

the questions[4], a relevant answer is retrieved but is not placed in the top 10 by the ranking algorithm. For these 28 questions in our set, re-ranking may be an option. Since re-ranking will not improve the results for the questions for which there is no relevant answer in the top-150, the maximum success@10 that we can achieve by re-ranking is 73.1% for answer paragraphs and 82.8% for documents.

5 Answer re-ranking

Before we can decide on our re-ranking approach, we take a closer look at the ranking method that is applied in the baseline system. The QAP algorithm includes the following variables: (1) term overlap between query and passage, (2) passage length and (3) total corpus frequency for each term (Buttcher et al., 2004). Let us consider three example questions from our collection to see the strengths and weaknesses of these variables.

1. Why do people sneeze?
2. Why do women live longer than men on average?
3. Why are mountain tops cold?

In (1), the corpus frequencies of the question terms *people* and *sneeze* ensure that the relatively unique term *sneeze* is weighted heavier for ranking than the very common noun *people*. This matches the goal of the query, which is finding an explanation for sneezing. However, in (2), the frequency variables used by QAP do not reflect the importance of the terms. Thus, *women*, *live*, *longer* and *average* are considered to be of equal importance, while obviously the latter term is only peripheral to the goal of the query. This cannot be derived from its corpus frequency, but may be inferred from its syntactic function in the question: an adverbial on sentence level. In (3), *mountain* and *tops* are interpreted as two distinct terms by the baseline system, whereas the interpretation of *mountain tops* as compound item is more appropriate.

Examples 2 and 3 above show that a question-answer pair may contain more information than is represented by the frequency variables implemented in the QAP algorithm. Our aim is to find out which features from a question-answer pair constitute the information that discloses a relation between the question and its answer. Moreover, we aim at weighting these features in such a way that we can optimize ranking performance.

[4] 73.1% − 43.0%

5.1 Features for re-ranking

As explained above, baseline ranking is based on term overlap. The features that we propose for re-ranking are also based on term overlap, but instead of considering all question content words indiscriminately in one overlap function, we select a subset of question terms for each of the re-ranking features. By defining different subsets based on syntactic functions and categories, we can investigate which syntactic features of the question, and which parts of the answer are most important for re-ranking.

The following subsections list the syntactic features that we consider. Each feature consists of two item sets: a set of question items and a set of answer items. The value that is assigned to a feature is a function of the intersection between these two sets. For a set of question items Q and a set of answer items A, the proportion P of their intersection is:

$$P = \frac{|Q \in A| + |A \in Q|}{|Q| + |A|} \quad (1)$$

Our approach to composing the set of features is described in subsections 5.1.1 to 5.1.4 below. We label the features using the letter f followed by a number so that we can back-reference to them.

5.1.1 The syntactic structure of the question

Example 2 in the previous section shows that some syntactic functions in the question may be more important than other functions. Since we do not know as yet which syntactic functions are the most important, we include both heads (f1) and modifiers (f2) as item sets. We also include the four main syntactic constituents for *why*-questions: subject (f4), main verb (f6), nominal predicate (f8) and direct object (f10) to be matched against the answer terms. For these features, we add a variant where as answer items only words/phrases with the same syntactic function are included (f5, f7, f9, f11).

Example 3 in the previous section exemplifies the potential relevance of noun phrases (f3).

5.1.2 The semantic structure of the question

The features f12 to f15 come from earlier data analyses that we performed. We saw that often there is a link between a specific part of the question and the title of the document in which the reference answer is found. For example, the answer to the question "Why did B.B. King name his guitar Lucille?" is in the Wikipedia article with the ti-

tle *B.B. King*. The answer document and the question apparently share the same topic (*B.B. King*). In analogy to linguistically motivated approaches to factoid QA (Ferret et al., 2002) we introduce the term *question focus* for this topic.

The focus is often the syntactic subject of the question. From our data, we found the following two exceptions to this general rule: (1) If the subject is semantically poor, the question focus is the (verbal or nominal) predicate: "Why do people sneeze?", and (2) in case of etymology questions (which cover about 10% of *why*-questions), the focus is the subject complement of the passive sentence: "Why are chicken wings called Buffalo Wings?" In all other cases, the question focus is the grammatical subject: 'Why do cats sleep so much?'

We include a feature (f13) for matching words from the question focus to words from the document title. We also add a feature (f12) for the relation between all question words and words from the document title, and a feature (f14) for the relation between question focus words and all answer words.

5.1.3 Synonyms

For each of the features f1 to f15, we add an alternative feature (f16 to f30) covering the set of all WordNet synonyms for all items in the original feature. Note that the original words are no longer included for these features; we only include the terms from their synonym sets. For synonyms, we apply a variant of equation 1 in which $|Q \in A|$ is interpreted as the number of question items that have at least one synonym in the set of answer items and $|A \in Q|$ as the number of answer items that occur in at least one of the synonym sets of the question items.

5.1.4 Cue words

Finally, we add a closed set of cue words that often occur in answers to *why*-questions[5] (f31).

5.2 Extracting feature values from the data

For the majority of features we need the syntactic structure of the input question, and for some of the features also of the answer. We experimented with two different parsers for these tasks: a development version of the Pelican parser[6] and the EP4IR dependency parser (Koster, 2003).

Given a question-answer pair and the parse trees of both question and answer, we extract values from each parser's output for all features in section 5.1 by means of a Perl script.

Our script has access to the following external components: A stop word list (see section 4.1), a fixed set of cue words, the CELEX Lemma lexicon (Burnage et al., 1990), all WordNet synonym sets, and a list of pronouns and semantically poor nouns[7].

Given one question-answer pair, the feature extraction script performs the following actions. Based on the question's parse tree, it extracts the subject, main verb, direct object (if present) and nominal predicate (if present) from the question. The script decides on question focus using the rules suggested in section 5.1.2. For the answer, it extracts the document title. From the parse trees created for the answer paragraph, it extracts all subjects, all verbs, all direct objects, and all nominal predicates.

For each feature, the script composes the required sets of question items and answer items. All items are lowercased and punctuation is removed. In multi-word items, spaces are replaced by underscores before stop words are removed from the question and the answer. Then the script calculates the proportion of the intersection of the two sets for each feature following equation 1[8].

Whether or not to lemmatize the items before matching them is open to debate. In the literature, there is some discussion on the benefit of lemmatization for information extraction (Bilotti et al., 2004). Lemmatization can be problematic in the case of proper names (which are not always recognizable by capitalization) and noun phrases that are fixed expressions such as *sailors of old*. Noun phrases are involved not only in the NP feature (f3), but also in our features involving subject, direct object, nominal predicate and question focus. Therefore, we decided only to lemmatize verbs (for features f6 and f7) in the current version of our system.

For each question-answer pair in our data set, we extract all feature values using our script. We

[5] These cue words come from earlier work that we did on the analysis of *why*-answers: *because, since, therefore, why, in order to, reason, reasons, due to, cause, caused, causing, called, named*

[6] The Pelican parser is a constituency parser that is currently being developed at Nijmegen University. See also *http://lands.let.ru.nl/projects/pelican/*

[7] These are the nouns *humans* and *people*

[8] A multi-word term is counted as one item

use three different settings for feature extraction: (1) feature extraction from gold standard constituency parse trees of the questions in accordance with the descriptive model of the Pelican parser[9]; (2) feature extraction from the constituency parse trees of the questions generated by Pelican[10]; and (3) feature extraction from automatically generated dependency parse trees from EP4IR.

Our training and testing method using the extracted feature values is explained in the next section.

5.3 Re-ranking method

As the starting point for re-ranking we run the baseline system on the complete set of 93 questions and retrieve 150 candidate answers per question, ranked by the QAP algorithm. As described in section 5.2, we use two different parsers. Of these, Pelican has a more detailed descriptive model and gives better accuracy (see section 6.3 on parser evaluation) but EP4IR is at present more robust for parsing long sentences and large amounts of text. Therefore, we parse all answers (93 times 150 paragraphs) with EP4IR only. The questions are parsed by both Pelican and EP4IR.

As presented in section 5.1, we have 31 re-ranking features. To these, we add the score that was assigned by QAP, which makes 32 features in total. We aim to weight the feature values in such a way that their contribution to the overall system performance is optimal. We set each feature weight as an integer between 0 and 10, which makes the number of possible weighting configurations 11^{32}. In order to choose the optimal configuration from this huge set of possible configurations, we use a genetic algorithm[11] (Goldberg and Holland, 1988). The variable that we optimize during training is MRR. We tune the feature weights over 100 generations of 1000 individuals. For evaluation, we apply cross valuation on five question folds: in five turns, we train the feature weights on four of the five folds and evaluate them on the fifth.

We use the feature values that come from the gold standard parse trees for training the feature weights, because the benefit of a syntactic item type can only be proved if the extraction of that item from the data is correct. At the testing stage, we re-rank the 93 questions using all three feature extraction settings: feature values extracted from gold standard parse trees, feature values extracted with Pelican and feature values extracted with EP4IR. We again regard the distribution of questions over the five folds: we re-rank the questions in fold five according to the weights found by training on folds one to four.

5.4 Results from re-ranking

Table 2 on the next page shows the results for the three feature extraction settings.

Using the Wilcoxon Signed-Rank Test we find that all three re-ranking conditions give significantly better results than the baseline ($Z = -1.91$, $P = 0.0281$ for paired reciprocal ranks). The differences between the three re-ranking conditions are, however, not significant[12].

5.5 Which features made the improvement?

If we plot the weights that were chosen for the features in the five folds, we see that for some features very different weights were chosen in the different folds. Apparently, for these features, the weight values do not generalize over the five folds. In order to only use reliable features, we only consider features that get similar weights over all five folds: their weight values have a standard deviation < 2 and an average weight > 0. We find that of the 32 features, 21 are reliable according to this definition. Five of these features make a substantial contribution to the re-ranking score (table 3). Behind each feature is its reference number from section 5.1 and its average weight on a scale of 0 to 10.

Moreover, there are three other features that to a limited extent contribute to the overall score (table 4).

Thirteen other reliable features get a weight < 1.5 assigned during training and thereby slightly contribute to the re-ranking score.

[9] Pelican aims at producing all possible parse trees for a given sentence. A linguist can then decide on the correct parse tree given the context. We created the gold standard for each question by manually selecting the correct parse tree from the parse trees generated by the parser.

[10] For this setting, we run the Pelican parser with the option of only giving one parse (the most likely according to Pelican) per question. As opposed to the gold standard setting, we do not perform manual selection of the correct parse.

[11] We chose to work with a genetic algorithm because we are mainly interested in feature selection and ranking. We are currently experimenting with Support Vector Machines (SVM) to see whether the results obtained from using the genetic algorithm are good enough for reliable feature selection.

[12] The slightly lower success and MRR scores for re-ranking with gold standard parse trees compared to Pelican parse trees can be explained by the absence of the gold standard for one question in our set.

Table 2: Re-ranking results for three different parser settings in terms of success@10, success@150 and MRR@150.

	Answer/paragraph retrieval			Document retrieval		
Version	S@10	S@150	MRR	S@10	S@150	MRR
Baseline	43.0%	73.1%	0.260	61.8%	82.8%	0.365
Re-ranking w/ gold standard parse trees	54.4%	73.1%	0.370	63.1%	82.8%	0.516
Re-ranking w/ Pelican parse trees	54.8%	73.1%	0.380	64.5%	82.8%	0.518
Re-ranking w/ EP4IR parse trees	53.8%	73.1%	0.349	63.4%	82.8%	0.493

Table 3: Features that substantially contribute to the re-ranking score, with their average weight

Question focus synonyms to doctitle (f28)	9.2
Question verb synonyms to answer verbs (f22)	9
Cue words (f31)	9
QAP	8.8
Question focus to doctitle (f13)	7.8

Table 4: Features that to a limited extent contribute to the re-ranking score, with their average weight

Question subject to answer subjects (f5)	2.2
Question nominal predicate synonyms (f23)	1.8
Question object synonyms to answer objects (f26)	1.8

6 Discussion

Our re-ranking method scores significantly better than the baseline, with use of a small subset of the 32 features. It reaches a success@10 score of 54.8% with an MRR@150 of 0.380 for answer retrieval. This compares to the MRR of 0.328 that Higashinaka and Isozaki found for *why*-QA and the MRR of 0.406 that Tiedemann reaches for syntactically enhanced factoid-QA (see section 2), showing that our method performs reasonable well. However, the MRR of 0.380 also shows that a substantial part of the problem of *why*-QA is still to be solved.

6.1 Error analysis

For analysis of our results, we counted for how many questions the ranking was improved, and for how many the ranking deteriorated. First of all, ranking remained equal for 35 questions (37.6%). 25 of these are the questions for which no relevant answer was retrieved by the baseline system at $n = 150$ (26.9% of questions). For these questions the ranking obviously remained equal (RR is 0) after re-ranking. For the other 10 questions for which ranking did not change, RR was 1 and remained 1. Apparently, re-ranking does not affect excellent rankings.

For two third (69%) of the remaining questions, ranking improved and for one third (31%), it deteriorated. There are eleven questions for which the reference answer was ranked in the top 10 by the baseline system but it drops out of the top 10 by re-ranking. On the other hand, there are 22 questions for which the reference answer enters the top 10 by re-ranking the answers, leading to an overall improvement in success@10.

If we take a look at the eleven questions for which the reference answer drops out of the top 10 by re-ranking, we see that these are all cases where there is no lexical overlap between the question focus and the document title. The importance of features 13 and 28 in the re-ranking weights works against the reference answer for these questions. Here are three examples (question focus as detected by the feature extraction script is underlined):

1. Why do neutral atoms have the same number of protons as electrons? (answer in 'Oxidation number')
2. Why do flies walk on food? (answer in 'Insect Habitat')
3. Why is Wisconsin called the Badger State? (answer in 'Wisconsin')

In example 1, the reference answer is outranked by answer paragraphs from documents with one of the words *neutral* and *atoms* in its title. In example 2, there is actually a semantic relation between the question focus (*flies*) and the document title (*insect*); however, this relation is not synonymy but hyperonymy and therefore not included in our re-ranking features. One could dispute the definition of question focus for etymology questions (example 3), but there are simply more cases where the subject complement of the question leads to document title than cases where its subject (such as *Winsconsin*) does.

6.2 Feature selection analysis

We think that the outcome of the feature selection (section 5.5) is very interesting. We are not surprised that the original score assigned by QAP is still important in the re-ranking module: the frequency variables apparently do provide useful information on the relevance of a candidate answer.

We also see that the presence of cue words (f31) gives useful information in re-ranking an-

swer paragraphs. In fact, incorporating the presence of cue words is a first step towards recognizing that a paragraph is potentially an answer to a *why*-question. We feel that identifying a paragraph as a potential answer is the most salient problem of *why*-QA, since answers cannot be recognized by simple semantic-syntactic units such as named entities as is the case for factoid QA. The current results show that surface patterns (the literal presence of items from a fixed set of cue words) are a first step in the direction of answer selection.

More interesting than the baseline score and cue words are the high average weights assigned to the features f13 and f28. These two features refer to the relation between question focus and document title. As explained in section 5.1.2, we already had the intuition that there is some relation between the question focus of a *why*-question and the document title. The high weights that are assigned to the question focus features show that our procedure for extracting question focus is reliable. The importance of question focus for *why*-QA is especially interesting because it is a question feature that is specific to *why*-questions and does not similarly apply to factoids or other question types. Moreover, the link from the question focus to the document title shows that Wikipedia as an answer source can provide QA systems with more information than a collection of plain texts without document structure does.

From the other features discussed in section 5.5, we learn that all four main question constituents contribute to the re-ranking score, but that synonyms of the main verb make the highest contribution (f22). Subject (f5), object (f26) and nominal predicate (f23) make a lower contribution. We suspect that this may be due to our decision to only lemmatize verbs, and not nouns (see section 5.2). It could be that since lemmatization leads to more matches, a feature can make a higher contribution if its items are lemmatized.

6.3 The quality of the syntactic descriptions

We already concluded in the previous section that our feature extraction module is very well capable of extracting the question focus, since f13 and f28 get assigned high weights by training. However, in the training stage, we used gold standard parse trees. In this section we evaluate the two automatic syntactic parsers Pelican and EP4IR, in order to be able to come up with fruitful suggestions for improving our system in the future.

As a measure for parser evaluation, we consider constituent extraction: how well do both parsers perform in identifying and delimiting the four main constituents from a *why*-question: subject, main verb, direct object and nominal predicate? As the gold standard for this experiment we use manually verified constituents that were extracted from the gold standard parse trees. We adapt our feature extraction script so that it prints each of the four constituents per question. Then we calculate the recall score for each parser for each constituent type.

Recall is the number of correctly identified constituents of a specific type divided by the total number of constituents of this type in the goldstandard parse tree. This total number is not exactly 93 for all constituent types: only 34 questions have a direct object in their main clause and 31 questions have a nominal predicate. The results of this exercise are in Table 5.

Table 5: Recall for constituent extraction (in %)

	subjs	verbs	objs	preds	all
Pelican	79.6	94.6	64.7	71.0	82.1
EP4IR	63.4	64.5	44.1	48.4	59.4

We find that over all constituent types, Pelican reaches significantly better recall scores than EP4IR ($Z = 5.57$; $P < 0.0001$ using the Wilcoxon Signed-Rank Test).

Although Pelican gives much better results on constituent extraction than EP4IR, the results on the re-ranking task do not differ significantly. The most plausible explanation for this is that the high accuracy of the Pelican parses is undone by the poor syntactic analysis on the answer side, which is in all settings performed by EP4IR.

7 Future directions

In section 4.3, we mentioned two directions for improving our pipeline system: improving retrieval and improving ranking. Recently we have been working on optimizing the retrieval module of our pipeline system by investigating the influence of different retrieval modules and passage segmentation strategies on the retrieval performance. This work has resulted in a better passage retrieval module in terms of success@150. Details on these experiments are in (Khalid and Verberne, 2008).

Moreover, we have been collecting a larger data collection in order to do make feature selection for

our re-ranking experiments more reliable and less depending on specific cases in our dataset. This work has resulted in a total set of 188 *why*-question answer pairs. We are currently using this data collection for further research into improving our pipeline system.

In the near future, we aim to investigate what type of information is needed for further improving our system for *why*-QA. With the addition of syntactic information our system reaches an MRR score of 0.380. This compares to the MRR scores reached by other syntactically enhanced QA systems (see section 2). However, an MRR of 0.380 also shows that a substantial part of the problem of *why*-QA is still to be solved. We are currently investigating what type information is needed for further system improvement.

Finally, we also plan experiments with a number of dependency parsers to be used instead of EP4IR for the syntactic analysis of the answer paragraphs. Current experiments with Charniak (Charniak, 2000) show better constituent extraction than with EP4IR. It is still to be seen whether this also influences the overall performance of our system.

8 Conclusion

We added a re-ranking step to an existing paragraph retrieval method for *why*-QA. For re-ranking, we took the score assigned to a question answer pair by the ranking algorithm QAP in the baseline system, and weighted it with a number of syntactic features. We experimented with 31 features and trained the feature weights on a set of 93 *why*-questions with 150 answers provided by the baseline system for each question. Feature values for training the weights for the 31 features were extracted from gold standard parse trees for each question answer pair.

We evaluated the feature weights on automatically parsed questions and answers, in five folds. We found a significant improvement over the baseline for both success@10 and MRR@150. The most important features were the baseline score, the presence of cue words, the question's main verb, and the relation between question focus and document title.

We think that, although syntactic information gives a significant improvement over baseline passage ranking, more improvement is still to be gained from other types of information. Investigating the type of information needed is part of our future directions.

References

Bilotti, M.W., B. Katz, and J. Lin. 2004. What works better for question answering: Stemming or morphological query expansion. *Proc. IR4QA at SIGIR 2004.*

Burnage, G., R.H. Baayen, R. Piepenbrock, and H. van Rijn. 1990. *CELEX: A Guide for Users.*

Buttcher, S., C.L.A. Clarke, and G.V. Cormack. 2004. Domain-Specific Synonym Expansion and Validation for Biomedical Information Retrieval.

Buttcher, S. 2007. The Wumpus Search Engine. http://www.wumpus-search.org/.

Charniak, E. 2000. A maximum-entropy-inspired parser. *ACM International Conference Proceeding Series*, 4:132–139.

Denoyer, L. and P. Gallinari. 2006. The Wikipedia XML corpus. *ACM SIGIR Forum*, 40(1):64–69.

Ferret, O., B. Grau, M. Hurault-Plantet, G. Illouz, L. Monceaux, I. Robba, and A. Vilnat. 2002. Finding an answer based on the recognition of the question focus. *Proc. of TREC 2001*, pages 500–250.

Goldberg, D.E. and J.H. Holland. 1988. Genetic Algorithms and Machine Learning. *Machine Learning*, 3(2):95–99.

Higashinaka, R. and H. Isozaki. 2008. Corpus-based Question Answering for why-Questions. In *Proc. of IJCNLP, vol.1*, pages 418–425.

Hovy, E.H., U. Hermjakob, and D. Ravichandran. 2002. A Question/Answer Typology with Surface Text Patterns. In *Proc. of HLT 2002.*

Khalid, M. and S. Verberne. 2008. Passage Retrieval for Question Answering using Sliding Windows. In *Proc. of IR4QA at COLING 2008.*

Koster, CHA. 2003. Head-modifier frames for everyone. *Proc. of SIGIR 2003*, page 466.

Quarteroni, S., A. Moschitti, S. Manandhar, and R. Basili. 2007. Advanced Structural Representations for Question Classification and Answer Reranking. In *Proc. of ECIR 2007*, volume 4425, pages 234–245.

Tiedemann, J. 2005. Improving passage retrieval in question answering using NLP. In *Proc. of EPIA 2005.*

Verberne, S., L. Boves, N. Oostdijk, and P.A. Coppen. 2008. Evaluating paragraph retrieval for *why*-QA. In *Proc. of ECIR 2008.*