Using Grammatical Relations, Answer Frequencies and the World Wide Web for TREC Question Answering

Sabine Buchholz*
ILK, Tilburg University
P.O. Box 90153, 5000 LE Tilburg
The Netherlands
s.buchholz@kub.nl

Abstract

This year, we participated for the first time in TREC, and entered two runs for the main task of the TREC 2001 question answering track. Both runs use a simple baseline component implemented especially for TREC, and a high-level NLP component (called Shapaqa) that uses various NLP tools developed earlier by our group. Shapaqa imposes many linguistic constraints on potential answers strings which results in not so many answers being found but those that are found have a reasonably high precision. The difference between the two runs is that the first applies Shapaqa to the TREC document collection directly, whereas the second one uses it on the World Wide Web (WWW). Answers found there are then mapped back to the TREC collection. The first run achieved a MRR of 0.122 under the strict evaluation (and 0.128 lenient), the second one 0.210 (0.234). We argue that the better performance is due to the much larger number of documents that Shapaqa-WWW's answers are based on.

1 Introduction

This paper presents the system that we used for the main task of the TREC-2001 question answering (QA) track. It differs from most of the other systems used in the QA track in the past years in three aspects. Firstly, it does not use a Named Entity recognition component. For example, it does not make a difference between persons and non-persons. Instead, the answer finding approach is centered on the main verb of the question. It relies on analyzing grammatical relations between this verb and the other parts of the question and answer. Secondly, it does not use complicated document, paragraph, sentence or string weights. The baseline component only counts the number of keywords in a sentence, and the Shapaqa component counts the frequency with which a given answer occurs. Thirdly and most importantly, it uses the World Wide Web as an additional resource.

2 System description

Our system does not have its own Information Retrieval engine. It uses the top ranked 1000 documents per question from the list provided by NIST.

There are two components: One high-level NLP component called Shapaqa[1] that has reasonable precision on the answer strings it returns but that often does not return any answer string,

*This research was done in the context of the "Induction of Linguistic Knowledge" research programme, which is funded by the Dutch Organization for Scientific Research (NWO).

[1] see also (Buchholz and Daelemans, 2001a), (Buchholz and Daelemans, 2001b) and http://ilk.kub.nl/shapaqa/ for an online prototype

and one simple keyword matching component that has low precision but nearly always returns some (possibly incorrect) answer string. The latter provides us with a baseline performance. The Shapaqa component and the baseline component are integrated as follows: If Shapaqa returns at least one answer string, its top answer string is taken as the top ranked answer string of the combined system. All the remaining ranks are filled by the top answer strings returned by the baseline system. We describe the two components separately in the following two sections.

2.1 Baseline

For the baseline component, the question is tokenized and then part-of-speech tagged (Daelemans et al., 1996). Then all those words are extracted as keywords that the POS tagger did not know (regardless of what tag it assigned) or that it tagged as noun, non-modal verb, adjective, particle number or foreign word except the words "much", "many", "name" and forms of "be", "have" and "do". If the question contains "did" or "does", all following infinitival verb forms are replaced by their past tense or third person singular present tense form, respectively. This rule applied to 22 and 14 questions, respectively, and uses the CELEX lexical database (Baayen, Piepenbrock, and van Rijn, 1993).

After keywords have been extracted, all top 1000 documents for each question (according to the list provided by NIST) are tokenized, and all sentences are extracted which contain at least one keyword for that question. Multiple occurrences of the same keyword are only counted once. The document ID and the number and position of the keywords are stored together with each extracted sentence. This yields 1,233,692 sentences. For two questions no sentences could be extracted.[2] The sentences for each question are then sorted according to the number of keywords in them.

The top sentences of the sorted list are taken as the baseline component's answers. To trim them to 50 bytes we compute that byte position in the sentence that is at the center of all the keyword positions, and then take 50 bytes around it (possibly shifting the 50 byte window as far as necessary to the right or left for it not to extend beyond the sentence boudary).

As an example, consider the question "When was President Kennedy shot?". Keywords are *President Kennedy* and *shot*. 5928 sentences are extracted. The topmost chosen answer sentence and its 50-byte string are "In 1963, he was riding in the motorcade wi*th President Kennedy when Kennedy was fatally shot* by Lee Harvey Oswald in Dallas.", in which the answer falls outside the chosen 50 bytes.

On the non-variant questions of the TREC-9 data, taking the full top five sentences of the sorted list as answers yielded an MRR of 0.321 using the automatic evaluation. After the sentences were trimmed to 50 bytes with the above method, MRR dropped to 0.125. This is hardly surprising, given that the method is completely insensitive to the type of answer the question is asking for (it returns the same 50 byte window, whether the question word is "who", "where", or "when" etc.) and that the complete answer sentences were on average 293 bytes long.

2.2 Shapaqa

2.2.1 Question Analysis

For the Shapaqa component, the POS tagged question is further chunked and grammatical relations between verb chunks and other chunks are determined (Buchholz, Veenstra, and Daelemans, 1999). Possible relations are subject (SBJ), object (OBJ), logical subject of passive verbs (LGS), locative adjunct (LOC), temporal adjunct (TMP), adjunct of purpose and reason (PRP), manner adjunct (MNR), and an unspecified relation between a verb and a prepositional chunk (OTH).[3] In the latter case, the preposition is stored as additional information. As the relation finder does

[2] "What is pilates?" and "What is dianetics?". Keyword matching is sensitive to capitalization and both terms appear in the documents only with capital letters.

[3] The relations are based on the functional tags in the Wall Street Journal corpus of the Penn Treebank II (Marcus et al., 1994) on which the tagger, chunker and relation finder are trained.

not work well on questions, especially in the first part, which shows the characteristic question syntax, a set of hand-made regular expressions and substitutions (developed on the TREC-9 data) are applied to it after parsing to fix the most common errors. For our previous example, the result would be "$[_{ADVP-TMP-1}$ When] was $[_{NP-SBJ-1}$ President Kennedy] $[_{VP-1}$ shot] ?".

Next, the parsed question is transformed into Shapaqa's internal format. Prepositions stranded at the end of the question are rejoined with their NP. Passive is converted to active. The head (i.e. the last word) of the first verb chunk is the central verb (auxiliaries in inverted questions should not be a verb chunk of their own). All the chunks that are directly dependent on it are stored together with their relation. Chunks that are not directly dependent on the central verb are concatenated to the next chunk to the left that is, because they are probably dependent on that chunk and thus, as relations seldom cross, belong to the same phrase. In this way the question is split up into phrases that all have some relation to the central verb. The phrase that contains the wh-word is replaced by a special marker ("?"). Our example would then look like: VERB="shot" OBJ="President Kennedy" TMP="?"

Questions asking for "Which/What X" are mapped as if they were simple "Who/What" questions. The phrase "In which state" was mapped as a simple "Where" because the relation finder assigned it a locative relation. In total, 60 "Which/What X" questions could be mapped. Questions starting with "Name a X" where mapped as if they read "What is a X?" (which is an ad-hoc solution which did not work: both questions were incorrectly answered). Some other minor simplifications are also performed during mapping.

In total, 416 questions could be converted to Shapaqa's internal format.

2.2.2 Answer extraction

Once the central verb and all its related phrases are in Shapaqa's internal format, they can be matched against parsed sentences extracted from the documents. As parsing is time-consuming, and Shapaqa phrases normally include all keywords (see Section 2.1), only those sentences are POS tagged, chunked and assigned relations that contain all the question's keywords (in total 44,753 sentences).

A parsed sentence matches the information in Shapaqa's internal format if the central verb matches with a verb chunk's head in the sentence, if all the phrases match literally somewhere in the sentence (the special marker "?" matches any chunk) and if each phrase's relation in the internal format matches the relation assigned to the chunk that contains the beginning of this phrase in the sentence (recall that a phrase's relation in the question is also determined by its first chunk's relation). Two relations match if they are identical, or if they form a special pair with special conditions . Special pairs are SBJ/OBJ where the former occurs in a passive question or sentence and the latter in an active one, LGS/SBJ under the same condition, and SBJ/OBJ if the verb is a form of "to be" (to match: "Who is X?" with "X is the president of Y")

If a sentence matches the internal format, we extract the chunk that matched the special marker "?" as an answer unless it is any of a fixed number of semantically empty answers like "he/who/somebody" etc.[4]

We isolate the head word of the answer chunk, i.e. its last word, and update a frequency counter for it. After all answer chunks have been extracted, they are sorted according to the frequency with which their head word was found as head of an answer .

2.3 Shapaqa-TREC versus Shapaqa-WWW

We used two versions of Shapaqa in our system. Shapaqa-TREC extracts answers directly from the TREC document collection in the way described above. One of the answer chunks whose head has the highest frequency is taken as the answer of the Shapaqa component. If necessary the answer is trimmed to 50 bytes by cutting off the end. For our example, the answer was extracted

[4] Obviously this does not work for questions whose answer is not a chunk but a clause (in that case only the complementizer chunk would be extracted).

from the sentence: "President Kennedy was shot *on Nov. 22, 1963.*" The head word was found three times as head of an answer.

Whereas Shapaqa-TREC extracts answers directly from the TREC document collection, Shapaqa-WWW first extracts answers from the World Wide Web. The central verb and the phrases of the question in Shapaqa's internal format are used as search terms for Google (http://www.google.com). Google returned results for 380 of the 416 questions that could be converted to Shapaqa's internal format. Shapaqa-WWW then searches for answers in Google's top 1000 text snippets. There, it found answers for 283 questions (7936 answers in total). For some questions, the only answers found are semantically empty ones like "he/who/somebody" etc., which are discarded. This leaves us with answers for 265 questions, which is slighty more than half of all TREC-10 questions. As explained above, answers are sorted according to the frequency of their head word. The most frequent head word is taken to be the preliminary answer. For our example, the head word *1963* was found twelve times.

To make a preliminary answer from the WWW into a valid TREC answer string, we have to find a document in the TREC collection that contains the head word. To increase the chance that the document actually supports the answer (as is necessary to be judged correct under the strict evaluation), we look for the head word in the sorted sentences extracted for the question, starting with the ones that contain most of the question keywords. Head words could be found in these sentences for 226 questions. After a sentence is found, a 50 byte piece of it centered around the head word (unless shifted to meet sentence boundaries) is extracted as Shapaqa-WWW's answer string. For our example, answer sentence and string are: "Ruby shot Oswald to death with the .38-caliber Colt Cobra revolver in the basement of Dallas *City Jail on Nov. 24, 1963, two days after Presi*dent Kennedy was assassinated." which unfortunately makes the answer invalid.

3 Runs: TilburgILKs and TilburgILK

We submitted two runs for the TREC QA track main task. Run TilburgILKs uses Shapaqa-WWW's answer (if present) on the first rank, Shapaqa-TREC's answer (if present) on the next highest rank, and the baseline component's answer on all other ranks. If not enough answers can be found to fill the five ranks, NIL (meaning that no answer exists in the document collection) is added.[5] We did not use the option to specify a "final answer" other than rank one.

TilburgILKs received a MRR of 0.210 (0.234 lenient). Run TilburgILK uses only Shapaqa-TREC (for the first rank) and the baseline component. Its MRR is 0.122 (0.128 lenient).

When writing this paper, we unfortunately noticed a serious bug in Shapaqa-TREC. The SBJ/OBJ special pair (see Section 2.2.2) was omitted in this implementation.[6] This means that for example for the frequent question type "What is a Y?" which asks for a definition or description , only sentences matched which read "X is a Y" and not those that read "A Y is X" (which is the way to formulate a definition). This led to many erroneous answers like "What is a prism?" – "Serbian agression", extracted from a sentence that reads "Serbian aggression is a prism through which we can see all sides of Europe."

As 254 of the questions that could be mapped to Shapaqa's internal format have a form where the special pair could have been applied, the omission might have influenced Shapaqa-TREC's performance significantly. To find out how serious this effect is, we reran the TilburgILK run after fixing the bug and studied the differences. It turned out that only 50 questions were affected by the bug fix, so we compared these manually. In most of the cases, the new answer was as bad as the old one, so the score did not change. We estimate that the new version's MRR would be about .004 higher than the original one's, which seems neglectable.

Table 1 displays the distribution of the rank of the first correct answer in both runs and shows that TilburgILKs has twice as many correct answers at the first rank than TilburgILK.

[5] Due to a bug, the NIL answer was not added at the lowest rank, as intended, but at the highest.

[6] Shapaqa-WWW is implented in PHP and runs on our webserver, whereas Shapaqa-TREC is in Perl and runs on a machine whose hard disk can hold all the TREC documents for the QA track. However, both systems access the same tagger, chunker and relation finder through socket connections.

rank of first correct	1	2	3	4	5	none
TilburgILK	43	15	15	7	14	398
TilburgILKs	87	19	9	7	11	359

Table 1: Distribution of rank of first correct answer in both submitted runs (strict evaluation)

components used	TilburgILKs	after bug fix
none (only answer NIL given)	3	3
only baseline	267	263
Shapaqa-TREC and baseline	4	8
Shapaqa-WWW and baseline	195	159
Shapaqa-WWW, Shapaqa-TREC and baseline	31	67

Table 2: Number of questions for which different components contributed to the five answer strings.

Table 2 shows how often each of the three components contributed to the five answers for the TilburgILKs run before and after the bug fix. We see that evene after the bug fix, Shapaqa-WWW was applied nearly three times as often as Shapaqa-TREC. This is probably due to the much larger document collection that Shapaqa-WWW works on.

Table 3 shows how the assessors judged each component's answer strings. We compute the *precision* of a component as the percentage of correct answers among all the answers it contributed. We see that Shapaqa-WWW has a higher precision than Shapaqa-TREC, especially after the bug fix. Both versions of Shapaqa have a much higher precision than the simple baseline component. Precision of the baseline component's answer is slightly higher for its top ranked answer than for the lower ones, but in general there does not seem to be a clear correlation between its answers' ranks and their reliability. Shapaqa-WWW has more unsupported answers than the other components. This is clearly due to the mapping from WWW-answers to TREC documents.

Table 4 gives a breakdown of Shapaqa-WWW's precision on different types of questions. The first column shows the question type according to the module that maps questions to Shapaqa's internal format. The second column indicates how many questions of this type could successfully be converted to Shapaqa's internal format. The third columns shows for how many questions Shapaqa-WWW returned an answer string. The fourth and fifth columns gives the precision on these strings (strict and lenient). "When"-questions do best, followed by those with the wh-word inside a prepositional phrase.

Shapaqa relies on high-level NLP (chunking, grammatical relations) for finding answers. If it finds more than one answer however, it uses the frequencies of the answers' head words to chose among the answers. The idea is that frequency correlates roughly with reliability of the answer. However, this can only work if the document collection from which answers are extracted is large enough. On average, Shapaqa-TREC (after the bug fix) finds 2.55 different answers (where "different" means having a different head word) for the questions it finds answers for at all. The average frequency of the most frequent answer for each question is 1.44, but the distribution is highly skewed in that most top answers have only frequency one, and only one top answer has a frequency higher than ten. By contrast, Shapaqa-WWW finds 17.0 different preliminary answers per question, and the average frequency of the most frequent answers is 26.7.

To further study the correlation between frequency and reliability, we divided the 265 questions for which Shapaqa-WWW found an preliminary answer into a high-frequency and a low-frequency group according to the frequency of the top answer (greater, or less or equal than 7). The precision of Shapaqa-WWW's answers for questions in the high-frequency group is 36.1% (39.3%) whereas it is only 26.0% (32.7%) for the low-frequency ones.

This shows that answers which we find often are more reliable than those with little evidence. As Shapaqa-WWW searches on a much larger document collection than Shapaqa-TREC, it can take advantage of this fact. This effect even holds despite the very simplistic way of mapping the WWW answers back to the TREC collection in order to comply with the TREC guidelines.

judgement	baseline 1st	2nd	3rd	S-TREC	S-TREC*	S-WWW
incorrect	462	473	469	27	63	144
correct	34	22	26	8	12	71
unsupported	1	1	1	0	0	11
prec. strict	6.8	4.4	5.2	22.9	16.0	31.4
prec. lenient	7.0	4.6	5.4	22.9	16.0	36.3

Table 3: Judgements (under strict evaluation) and precision of answers (strict and lenient) of each component. Baseline 1st, 2nd and 3rd means the baseline's first, second and third answer. S-TREC is Shapaqa-TREC, S-TREC* is Shapaqa-TREC after the bug fix, S-WWW is Shapaqa-WWW.

Wh-type	# of questions	# of q. answered	prec. strict	prec. lenient
who	44	21	57.1	57.1
what	237	136	25	30.1
which X	5	2	0	0
what X	55	27	18.5	25.9
where	24	12	25	25
when	24	18	66.7	72.2
why	2	0	0	0
how	7	1	0	0
PP	16	9	55.6	66.7
Name a	2	0	0	0

Table 4: Precision of Shapaqa-WWW on different wh-types.

4 Error analysis

In this section, we study the effect of several design decisions we made about the document selection, the transformation from natural lanaguage questions to Shapaqa's internal format and the mapping from WWW preliminary answers back to the TREC collection.

Our system does not search the whole document collection for answers but only the top 1000 documents per question (as provided by NIST). For 14 questions, some other systems found an answer in documents not in the top 1000, so some minor improvement should be possible through a better IR component.

Several things can go wrong during the transformation from natural language questions to Shapaqa's internal format. First, if no central verb can be found, the transformation is not possible. This happened with 14 questions. In three of these cases there really was no verb (e.g. "How many liters in a gallon?"), so the system would have needed a special rule to insert the verb "are". In one case the main verb "was" was not analyzed as a verb chunk. In the other ten cases, the main verb is analyzed as a noun. This problem is probably due to too few questions in the training material of the tagger.

Second, two chunks are assigned the same relation and there is no coordinating conjunction between them. This happened with 22 questions. Sometimes the chunks really have the same relation (e.g. "*In Poland*, *where* do most people live?"), sometimes the analysis is due to the relation finder's failure to distinguish between different object relations ("*What* do you call *a newborn kangaroo*?"), but most of the times the analysis is just plain wrong. Again, more questions in the training data could improve performance.

Third, some chunk has a direct relation to the central verb that does not fit any of the predefined categories (28 cases). These are mostly nouns mistagged as adverbs which then give rise to adverbial chunks being neither locative nor temporal, manner or purpose/reason. In some cases they are adjectival complements of "to be". The system needs to be extended to deal with these categories.

Fourth, questions with "How many/How much" or "How X" where X is some adjective cannot

be converted (12 cases). Finally, there are some rare cases, like no relation between the *wh*-phrase and the central verb or failure to recognize the question phrase as such.

Shapaqa treats "which/what X" questions like simple "who/what" questions. To see in how far this influences performance, we manually checked the top frequency answers found on the WWW for 10 of these questions. In three cases the topmost answer looks okay. In two other cases at least the second answer is correct (e.g. "What river in the US is known as the Big Muddy?" – "The river"; "The Missouri"). For the remaining five questions, the missing constraint leads to wrong results ("Which president was unmarried?" – "The mother"). So ideally, we would want to use the extra information. However, we would then need a component that can e.g. decide whether something is a "mountain range in North America". Until that time, our solution is an approximation.

We conducted a similar survey to find out how harmful our approach of ignoring the difference between *who* and *what* is. A manual check on Shapaqa-WWW's answers for five questions of each type suggests that this simplification does not introduce many errors. This makes sense, given that questions like "Who/What discovered radium?" are very unlikely to have any "what" answers, and questions like "Who/What is Australia's national flower?" are unlikely to have any "who" answers. The only counterexample is "Who developed the Macintosh computer?", for which the assessors did not accept "Apple Computer Inc." or similar.

Clearly, the forced mapping from preliminary WWW answers to TREC documents is far from ideal. Sometimes Shapaqa-WWW finds the correct answer on the WWW but cannot map it. One reason is that no answer exists in the collection. This happened with "What is Australia's national flower?" – "The Golden Wattle". Alternatively, the answer that came up highest from the WWW does not exist in the collection but others do. This happened with "What is a shaman?" and the answer "a healer" (other systems found answers like "tribal magician" or "a kind of priest").

A problem for both versions of Shapaqa are question-answer-pairs like "What is mold?" – "Mold is a problem". This answers the question in letter but not in spirit. There is probably a limited number of abstract nouns that can occur in this contruction (another one is "solution") and explicitly excluding the most common ones might be an opportunistic solution. A more principled approach would probably need semantic knowledge.

"What is a panic disorder?" – "A panic disorder is a type of generalized anxiety disorder." Although this answer as a whole is okay, Shapaqa identifies "type" as the head of the (predicative) object, so it is this word that gets looked for in the TREC document sentences, which might or might not work. As in the previous case, there are probably only a limited number of nouns that cause this problem but a general solution needs semantic knowledge.

"What is epilepsy?" – "Epilepsy is a common neurological disorder." This answer is correct and the head is correctly identified, too. However, as the head is rather unspecific, the answer string that is finally extracted from the TREC documents ("from an inner-ear disorder that causes vertigo") makes the answer invalid. In contract to the previous cases, this problem is entirely due to the forced mapping from WWW answers to TREC documents, so its solution is not of general interest. One might try to find a match not only for the head word but also for the other words in the chunk.

5 Summary

We described our approach to QA which combines a basic keyword matching component with a high-level NLP component that uses chunking and grammtical relations to impose many linguistically motivated constraints on what it extracts as an answer. This results in a much higher precision but less answers. We tackle the problem by searching for answers not only in the TREC document collection but also on the WWW. This results in answers to more questions and more reliability of the answers through the use of answer frequencies. Many aspects of the system can still be improved, but the achieved MRR of 0.210 (0.234) is certainly encouraging.

References

Baayen, R. H., R. Piepenbrock, and H. van Rijn. 1993. *The CELEX lexical data base on CD-ROM.* Linguistic Data Consortium, Philadelphia, PA.

Buchholz, Sabine and Walter Daelemans. 2001a. Complex answers: A case study using a www question answering system. *Journal of Natural Language Engineering*, Special issue on question answering. To appear.

Buchholz, Sabine and Walter Daelemans. 2001b. SHAPAQA: Shallow parsing for question answering on the world wide web. In *Proceedings of RANLP - 2001.*

Buchholz, Sabine, Jorn Veenstra, and Walter Daelemans. 1999. Cascaded grammatical relation assignment. In Pascale Fung and Joe Zhou, editors, *Proceedings of EMNLP/VLC-99*, pages 239–246. ACL.

Daelemans, W., J. Zavrel, P. Berck, and S. Gillis. 1996. MBT: A memory-based part of speech tagger generator. In E. Ejerhed and I. Dagan, editors, *Proc. of Fourth Workshop on Very Large Corpora*, pages 14–27. ACL SIGDAT.

Marcus, Mitchell, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The penn treebank: Annotating predicate argument structure. In *Proceedings of ARPA Human Technology Workshop*, pages 110–115.