Goal-Driven Answer Extraction

Michael Laszlo, Leila Kosseim, and Guy Lapalme
RALI, DIRO, Université de Montréal
CP 6128, Succ. Centre Ville, Montréal (Québec) Canada, H3C 3J7
{laszlom, kosseim, lapalme}@iro.umontreal.ca

Abstract

We describe the structure and functioning of an answer-extraction system built from the ground up, in only three man-months, using shallow text-processing techniques. Underlying these techniques is the attribution to each question of a *goal type* serving to characterize the outward form of candidate answers. The goal type is used as a filter during long-answer extraction, essentially a small-scale IR process which returns 250-byte windows rather than whole documents. To obtain short answers, strings matching the goal type are extracted from these windows and ranked by heuristics. TREC-9 performance figures show that our system has difficulty dealing with brief, definition-based questions of the kind most likely to be posed by users. We propose that specialized QA strategies be developed to handle such cases.

1 System Overview

In the following we shall refer to our system by the working title XR^3, which stands for *eXtraction de Réponses Rapide et Robuste*, or "fast and robust answer extraction". This choice of name reflects the idea that in QA, answers are not known beforehand, but are sought in raw text with the aid of hints supplied by the user. That, at any rate, was the assumption which arose from our acquaintance with the TREC-8 test set; more recent experience with the TREC-9 questions suggests that answer generation from a knowledge base will indeed be an indispensable component of future QA systems.

XR^3 is composed of two distinct halves: the first returns long answers consisting of exactly 250 characters, while the second extracts short answers consisting of no more than 50 characters. The unified view in Figure 1 shows that the second round of processing recycles output from the first, and that they share certain intermediate results.

Input to XR^3 consists of a question and a small collection of source documents. Regardless of whether these documents are paired *a priori* with the question or identified on the fly, they are expected to contain a valid answer with high likelihood. For the purpose at hand, we used Amit Singhal's lists of relevant documents (as determined by an IR process which used the question as a query). We deem these to be of sufficient quality in light of our finding that 97% of the top-200 documents per TREC-8 question contain at least one valid answer. Time constraints prevented us from implementing our own IR process, but for the sake of broad coverage we used all 1000 documents per question issued for TREC-9.

A pattern-matching analysis of the question identifies a set of *keyterms* (keywords and keyphrases), a goal, and possibly a date, all of which are used in the extraction and scoring of 250-character windows. The best five of these may be directly submitted in the long-answer category, and that is exactly what we did for TREC-9. Internally, we refer to this as the *IR run*, since our scoring heuristic employs as a primary criterion the occurrence of keyterms. The best 100 of these windows are funneled to the second portion of XR^3, although not all of them will be used. Depending on the value of a system parameter, 10 to 20 are skimmed off the

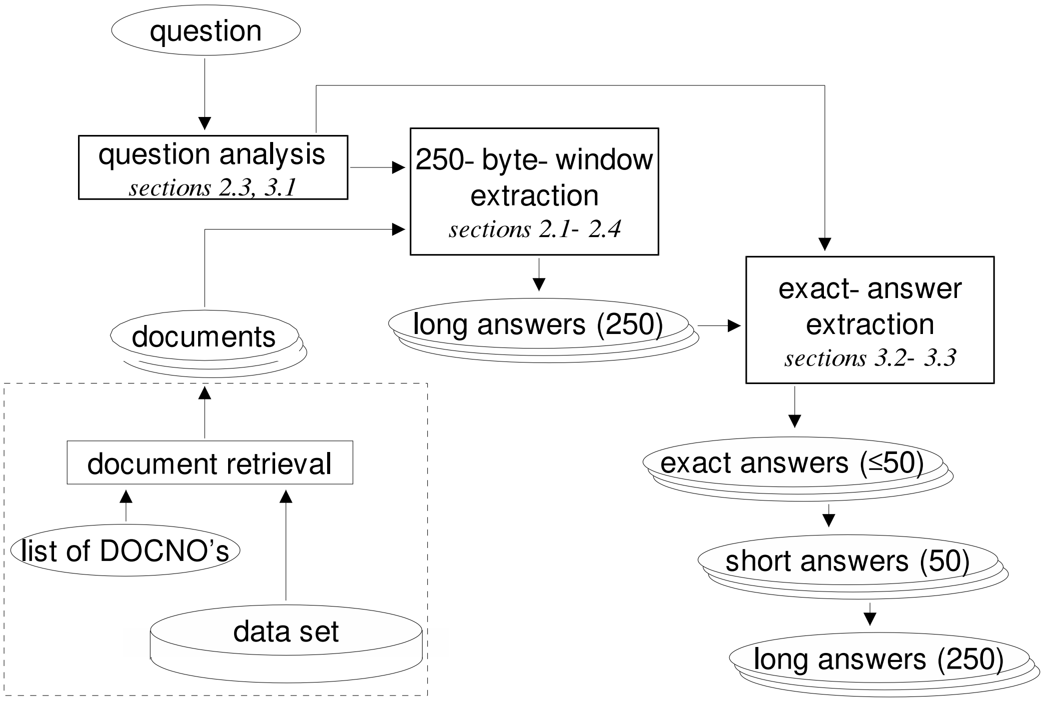

Figure 1: Process flow in XR^3: short answers are extracted from long answers; some results of question analysis are shared.

top (we settled on 10 prior to the TREC-9 runs) and, in a crucial application of the question goal which was identified earlier, exact answers are extracted. These strings are guaranteed to contain no more than 50 characters, but are typically a fifth that long, being precise expressions which name "Mount Everest" and "four miles" and other things considered to be potential answers. In any event, they are ripe for submission in the short-answer category. Expanding the string to a full 50 characters, and then to 250 characters, yields a second run in each category. In sum, then, we submitted four runs to TREC-9: two in the long-answer category, and two in the short-answer category (although one of these consists of our exact answers).

2 Long Answers

Let us note that the needs of QA are nearly satisfied by information retrieval systems, but for two shortcomings. The sheer size of a document returned by IR means that it is unlikely to fall within reasonable estimates of what constitutes an answer. And then there is no guarantee that this document will be relevant to some question on the user's mind, nor does the IR task require that a document be of any value to the user.

The quickest way around the first of these obstacles is to abandon orthodox conceptions of the document as an intact, coherent narrative. If we redefine the "document" as a string found in some portion of the corpus and not exceeding, say, 250 characters in length, then an IR system would, formally, also be a QA system. The second difficulty is the thornier one, but it is safe to say that if we were in possession of a function which maps questions to useful queries — if, that is, we had some idea of what words and phrases tend to occur in the vicinity of the answer to a given question — then IR alone would be adequate to the QA task.

These are the very principles by which our system extracts long answers. The first portion of XR^3 may be described as a lightweight IR engine adapted to the requirements of QA, having been furnished with a front-end utility which translates questions into queries and with shallow heuristics at the back end for scoring its results.

2.1 Windowing

We henceforth consider the data set not primarily as a collection of documents, but as a series of text *windows*, which are defined in XR^3 as follows. Given a list of keyterms, a window is any sequence of 250 characters found within a single document and containing a keyterm as near its center as possible. Of course, the focal keyterm should be precisely centered in all but the extremal windows, comprising the first and last 250 characters of a document, where it may be situated to the left or right of center, respectively. Apart from these two special cases, which may take hits from several keyterms, no window should be extracted more than once under our regime. In a more pragmatic sense, however, we are bound to end up with a great deal of redundancy. A non-extremal cluster of keyterm occurrences in the source text will result in the extraction of several windows that differ from one another by only a few words: these windows are semantically more or less equivalent, and would tend to receive identical scores.

The prospect of having our final list of top 50 or top 5 windows cluttered by superfluous windows is a disconcerting one, so we impose a limit on the amount of overlap permitted between any two windows in the form of a system parameter, called the maximum windowing margin, which is set at the beginning of a run and stays constant thereafter. With a margin of 0.1, for instance, we would take each cluster of windows whose offsets (distances from the beginning of the document) vary by fewer than $0.1 \times 250 = 25$ characters, arbitrarily choose one, and discard the rest.

We have been unable to settle on an optimal value for this margin, nor is it clear that there is one at all. It stands to reason that a very restrictive margin value such as 0.1 leads to the loss of viable windows, while high values such as 0.9 are insufficiently discriminating. But for the broad middle range between 0.2 and 0.8, the give-and-take of these two opposing trends, along with the perturbations resulting from the variation of other system parameters, make it impossible to arrive at an informed decision. Seeing this, we resolved to hold the margin constant at the unexceptionable value of 0.5, without pausing to wonder how many valid answers were falling through the cracks.

2.2 Keyterm Extraction

The process by which XR^3 assembles a query for its IR phase stands on the premise that words appearing in a question should, in the source text, be found in proximity to a valid answer. Our approach may be unique in that selected question tokens are used verbatim in the search query, without taking into account synonymy or even morphology. Our aversion to the use of synonyms issues at least in part from the report of the University of Ottawa's team at TREC-8, which notes that the results of their own brute-force QA system suffered from synonym sensitivity [ML99]. Furthermore, synonym occurrence is not necessarily valuable in early-phase extraction. The intuitive argument in favor of synonymy is that verbs and adverbs used in relation to an answer string are far more susceptible to variation than are the nouns, and certainly the proper nouns, found in a question. But expanding the IR query with sense-related words is likely to result in a large number of extraneous windows: if a question asks for the identity of someone who "won" the Nobel Prize, then the presence within a window of such words as "acquire", "gain", and "accomplish" may be misleading rather than useful.

Precisely because verbs and adverbs found in the question are subject to such a wide variety of rephrasings, we ignore them altogether, presuming that it is sufficient to collect all windows which make reference to, say, the Nobel Prize. In general, we regard the presence of some noun in the question as a qualifying condition for the presence of an answer. With that, we lose much of the need for stemming or morphological variation. Once those tokens tagged as determinants and prepositions and so forth have been discarded, only nouns and proper nouns, these latter having been identified on the basis of capitalization, are admitted into the search

goal type	occurrences
PNOUN	124
TIMEPOINT	23
CARDINAL	22
UNKNOWN	13
LINEAR	7
MONEY	6
PERCENTAGE	2
TIMESPAN	1
MEANS	1
REASON	1
AREA	0
VOLUME	0
MASS	0

Table 1: TREC-8 goal frequencies — number of times each type of goal was identified by the final version of XR^3 among questions 1-200 of the TREC-8 test set.

query. A second pass over the question adds groupings of adjectives, participles, and nouns, in order to capture such potentially useful noun phrases as “world energy output” and “managing director”.

2.3 Goal Identification and Extraction

A further condition windows should meet is that they each contain at least one expression corresponding to the *goal* of the question, which we understand as a loose characterization of the answer. For example, if it is clear from the form of a question that one is being asked to name a river — perhaps because it reads “Name a river which ...” — then we accept no window unless it contains something that could conceivably be the name of a river, such as a proper noun. Thirteen goal types are currently distinguished by XR^3, as listed in Table 1. A goal should have the dual virtues of being easily identified in the question and easy to extract from source text. Identification is accomplished by a series of regular expressions which rely on the presence of trigger words (“Who ...” → PNOUN) and patterns (“How much ... earn/spend/cost/...” → MONEY). If a question runs the gauntlet without drawing fire, the goal type is assumed by default to be PNOUN, which we take to be a fairly safe bet for questions of the type seen in TREC-8.

Extraction from the source text of strings matching the goal is again performed by regular expressions. Hand-crafted patterns and trigger words, such as lists of unit names associated with MONEY or with TIMESPAN, are applied throughout, except for the extraction of proper nouns. Here we apply a novel method which begins by extracting all capitalized words and groups thereof appearing beyond the first word of a sentence. The less trivial cases, where we must consider the very first word of a sentence, are resolved by comparing that word to known prepositions, or, in the extremity, by comparing it to proper nouns that have already been identified in the middle of a sentence.

As a final criterion, windows must match the date (if any) specified by the question. This highly specialized tactic was developed in response to those 11% of questions from the TREC-8 test set which named a specific year, as in *Which team won the Super Bowl in 1968?*(16).[1] In order for a window to be eligible under these circumstances, it must either contain the specified date (permitting some variation, such as ’68 for 1968) or it must be drawn from a document published on that date.

[1]After each question we parenthetically indicate its number, from 1 to 893, in the conventional TREC order.

score	recall	group bonus	type factor	date factor
.527	91%	0	1	1
.529	91%	0	1	10
.557	93%	0	10	1
.562	90%	1	1	1
.562	91%	1	1	10
.563	93%	0	10	10
.583	93%	1	10	1
.592	93%	1	10	10

Table 2: Parameter optimization — automatic evaluation on runs obtained by varying values of three key scoring parameters; performed only on questions 1-100 of the TREC-8 test set.

2.4 Scoring Long Answers

The answer-bearing potential of a window is measured first of all by the number of keyterms it contains — or rather by the variety of keyterms, since we do not take into account multiple occurrences of the same term. Weight values may range from 0 (so that the term is effectively ignored) or 1 (no greater weight than ordinary terms) to arbitrarily high values such as 20 (which would mean that such terms are worth as much as 20 ordinary ones). The sum obtained from the linear combination of keyterm hits (0 or 1) with their respective weights is then multiplied by some constant if the window was determined to suit the question goal, and by a final constant awarded for date correspondence. To be quite precise, the score is calculated as follows. Given keyterms $K_1, \ldots, K_n$, we know of two corresponding sequences: $c_1, \ldots, c_n$ where c_i is `capBonus` if K_i is capitalized, 1 otherwise; and $g_1, \ldots, g_n$ where g_i is `groupBonus` if K_i is a phrase, 1 otherwise. The function $f(K_i)$ returns 1 if K_i is present in the window under consideration, 0 otherwise. The partial score is given by

$$\sum_{i=1}^{n} f(K_i) c_i g_i$$

This score is then multiplied by a goal factor and a date factor as necessary. Experiments show that it is worthwhile to set these factors to a value high enough so that windows not meeting the goal and date criteria effectively drop out of sight in the final ranking. Increasing both factors from 1 to 10 improves the TREC score by a margin of 0.03. This gain, if not spectacular, is significant and strikingly consistent regardless of fluctuation in system parameters.

2.5 Results of Long-Answer Extraction

In making the quantitative observations above, we rely on TREC scores determined by an automatic evaluation which applies the extended set of Perl patterns derived from last year's judgments, producing an inflated estimate of the true score, *i.e.,* that which would be awarded by a human jury. Once system parameters have been fully tuned, XR^3 achieves a score of 0.511 on questions 1–200, which compares favorably to the results posted by last year's participants. We do, of course, possess the considerable advantage of having seen the questions beforehand. However, let it be noted that all pattern-matching rules were written for the first 100 questions, and all system parameters were optimized for the same set. Evaluation on the remaining questions gave us no reason to make changes of any significance to the IR portion of XR^3. Furthermore, automatic evaluation consistently yields TREC scores over 0.5 on any subset of questions save for the last 25, where performance suffers badly, dropping well below 0.4. We attribute this hiccup to the far greater brevity of the last 25 questions, which contain only 3.2 keyterms on average, compared to 5.7 for the set as a whole.

link type	example
`existence`	*Who is the Queen of Holland?*(136)
`attribute`	*How tall is the Matterhorn?*(161)
`time`	*When was Yemen reunified?*(130)
`location`	*Where is Inoco based?*(20)
`reason`	*Why are electric cars less efficient in the north-east than in California?*(159)
`means`	*How did Socrates die?*(198)

Table 3: Link types — the six types of question link identified by XR^3, with a characteristic example of each; the link is understood to be a relationship between the named focus and an unnamed answer.

3 Exact Answers

We go to the trouble of extracting self-contained answers from a shortlist of 250-character windows not only for the sake of submitting them in a new category, but with the hope of improving performance in the old one. It turns out that the prospects for finding exact answers are fairly bright, considering the density with which valid answers occur near the top of our long-answer list. The first five windows contain at least one valid answer about two-thirds of the time according to our automatic evaluation on the TREC-8 test set, while the number-one window alone is hit with 40% probability.

At the heart of our exact-answer extraction process is the assignment to each question of a goal, which previously played so marginal a role in window filtering. Now, however, for a given set of windows we know of a set of substrings which are of the form required by the question under consideration. The goal identification and extraction processes were designed very conservatively, emphasizing recall rather than precision; thus, we can state with reasonable confidence that if any one of the windows contains a valid answer, then one of our strings will constitute a valid answer.

3.1 Question Analysis

We have forsaken the computationally expensive and mostly impossible task of formal parsing in favor of something more modest, something that captures less information but stands a better chance of being fully realized. Our pet theory this year was that all questions explicitly describe a relationship between something unknown — to wit, the answer — and something quite concrete. The "something unknown" is already described by our goal, so it remains to identify the *link* and the *focus*. To us, the focus of a question is that portion of it which must absolutely be found within a window, whether verbatim or in some variant form, if we are to believe that this window can contain a valid answer. It is not clear whether every question contains such an expression, but we will proceed on the assumption that it is there. Given the question *What was the monetary value of the Nobel Peace Prize in 1989?*(2), for instance, the focus should be identified as "Nobel Peace Prize"; in the source text, we would then look for strings bearing some lexical resemblance to it.

A question's link is not so much identified as interpreted from its syntactical structure. Of the six possible link types (see Table 3), the one most relevant to the above question is presumably `existence` — that, at any rate, is the truth according to XR^3. It might be argued that `attribute` would be a more intuitively correct interpretation, but such quibbles are beside the point. The utility of a question analysis is sharply limited by the accuracy of the source-text analysis against which we intend to compare it, and as things stand, this latter is so much more difficult a task that even our very rough characterization of the question will suffice.

The link and focus are both determined through pattern matching. We view the question as a sequence of phrases hinging on stopwords, which are here defined as prepositions, determinants,

question	components	content of window
Which team won the Super Bowl in 1968?(116)	*pre*	terback, Namath, one of the first since Babe Ruth to make a fortune playing a game. With Namath as their leader, the AFL's 1968
	answer	New York Jets
	post	went into
	focus	Super Bowl
	right	III as an 18-point underdog and won, 16-7, against the NFL champion Baltimore Colts, wh
How many people live in the Falklands?(101)	*left*	everything. If they bought someone out, where would they go? Mr Di Tella denied that payments would be made to encourage people to leave the
	focus	Falkland Islands
	pre	and settle elsewhere. He said, 'We want to be very respectful of these
	answer	2,000
	post	people. They have liv

Table 4: Schematization of candidate windows — the text of each 250-character window is broken up into five named fields, here separated by carriage returns; note that in the first case, the answer occurs earlier than the focus, but the contrary is true in the second case.

pronouns, and common verbs such as "is" and "does". These stopwords are perceived as falling naturally into certain patterns which betray the semantic relationships between the adjoined phrases.

3.2 Window Schematization

Once XR^3 has decomposed a question with the aid of stopwords and so inferred its goal, link, and focus, it attempts to unify this description of the question with syntactical structures found in the source text. We assume that all viable windows will contain something resembling the question focus (which we shall simply call a "focus" in the interest of brevity). Furthermore, the window must contain at least one candidate answer (hereafter, an "answer") — an expression matching the question's goal. We then take the string lying between the focus and the answer, and try to ascertain whether it is related to the question link. That is not to say that the remaining portions of the window are of no interest; nonetheless, we restrict our attention to this particular string for the sake of simplicity and computational tractability.

In order to carry out such an assessment, XR^3 considers every possible focus-answer pairing found in a given window, and for each of these the window is subdivided into five components in one of the two ways illustrated in Table 4. If the answer is to the left of the focus, the appropriate scheme is [*pre, answer, post (inter), focus, right*], whereas the converse calls for [*left, answer, pre (inter), focus, post*], where *pre* and *post* are understood in relation to the answer.

3.3 Scoring Exact Answers

The overall score awarded to an answer and its attendant apparatus is a linear combination of three partial scores. The first consists of the IR score previously determined for the window from which this answer was extracted. The second component, our *link score*, is calculated by one of six groups of regular expressions according to the value of the question link. These are triggered by patterns which have proven, in our experience, to be strongly associated with the link. In the case of existential links, one of the more prominent trends involves constructions of

run	score
long1	.511
exact	.331
short	.386
long2	.524

Table 5: Best results obtained by automatic evaluation on TREC-8 questions 1-200.

the form *answer,* ***a/an/the*** *...focus*, as in "Colin Powell, an American general" or "Everest, the world's highest peak". Thus, we reward those answer schemes where the connecting string begins with a comma and a determinant.

The third component is something called the *extra score*, which is determined by ancillary considerations. Foremost among these is a special treatment for those questions whose goal has been determined to be some number of specific units, such as `CARD(calories)`. Also taken into consideration are the presence of words found in the question but not in the focus (as a precaution against focusing error) and additional hits scored by a simple stemming mechanism based on regular expressions.

3.4 Answer Expansion

Once all the goal-type expressions extracted for a given question have been scored and ranked we can offer the top five as exact answers, even though there is no such category in the TREC-9 evaluation. These strings tend to be rather shorter than the 50 characters allowed in the short-answer category — about 10 characters long — so we may easily cast our net a little wider by adding characters to the left and right until the limit is reached. The strings which result from such an expansion tend to suffer from the same redundancy problem experienced in the first phase of long-answer extraction: very often, several 50-character windows will end up covering the same cluster of exact answers. Again, XR^3 makes use of a maximum windowing margin to discard superfluous answers. This is done without disturbing the order of the remaining answers, so that they remain ranked in a useful way, namely, that determined by the exact-answer score. The top five may then be offered as a second submission in the short-answer category.

The expansion to long answers is accomplished with even greater ease, since we have retained the 250-character windows from which the exact answers were extracted. Each of these is now awarded the score of the corresponding exact answer, the entire collection is sorted, redundancies are cast out, and the result is an ostensibly improved list of long answers.

3.5 Results of Short-Answer Extraction

The limited selection of scoring criteria described above was distilled from a far broader field. Most speculative heuristics fell by the wayside as they failed to boost scores in the context of XR^3's performance on the TREC-8 test set. Among these were functions which measured resemblance between the designated focus and the candidate focus; calculated the average distance of all keyterms from the candidate answer; and employed a more comprehensive stemming algorithm than the one currently in use. In the end, we arrive at the scores displayed in Table 5. Recall that the first run was produced by means of our 250-character windowing mechanism, while the remaining three all derive from exact-answer extraction.

Our score of 0.331 for exact answers seems to be a decent performance: it would, in theory, have ranked fourth out of the 18 short-answer runs at TREC-8. Note that these exact answers were extracted from only the 10 best long answers. Any more than that and scores suffer, presumably because the scoring algorithm does fairly well at reshuffling high-quality windows but is too naive to recognize low-ranking junk for what it is. Using only the IR score to rank

run	TREC-9 assessment	+/− from mean	% from mean	mean over all runs
long1	.352	+.002	+.01%	.350
exact	.149	−.069	−32%	.218
short	.179	−.039	−18%	.218
long2	.366	+.016	+4.6%	.350

Table 6: Official results achieved on questions 201-893 of the TREC-9 set, with comparison to arithmetic mean of all runs submitted in the respective category (50 or 250 characters). Recall that our exact answers are 10 characters long on average, and that the first run is our IR run.

exact answers, rather than the entire tripartite formula, results in a not much worse TREC score of 0.237, demonstrating the pervasive influence of brute-force IR.

Sheer chance also has a role to play, given that the expansion to 50 characters results in a markedly improved TREC score of .386. Final expansion to 250-character windows leads to an expected — but unexpectedly small — improvement over the first run, from .511 to .524. Surprisingly, these TREC scores remain fairly even across the entire range of goal types, whereas one might have expected the frequent and coarse `PNOUN` extraction to fare poorly against the more specialized types, such as `MONEY` and `SPECTIME`.

4 Evaluation

4.1 TREC-9 Performance

The first column of figures in Table 6 gives the "strict" TREC-9 assessment of each of our runs, while the last column indicates the mean of the TREC scores assessed over all runs in each category. Clearly, XR^3 has suffered a general and sizeable decrease in performance — this year's long answers result in a lower score than last year's short answers. We attribute this in part to the many features of this year's test set which were not present at TREC-8.

Questions 201 through 893 tend be much shorter than the first 200, offering 3.8 keyterms on average as opposed to 5.7 with the earlier set. In addition, a notable number are of the form *What is* X*?*, whereas these were entirely absent from the TREC-8 set, with the exception of *What is Head Start?*(115). It is not clear that such requests for information as *Who is Anubis?*(222) and *Where is the Danube?*(226) fit the description of "short-answer, fact-based questions," as the QA Track Guidelines would have it. What is clear, however, is that these are precisely the sorts of questions that users want to ask and do end up asking today, even in queries submitted to commercial search engines.

We noted an unusual number of misspellings among the questions: *Superbowl*, *Nicholas Cage*, *applicances*, and *Pittsburg*, for instance, are all erroneous, and a system which made no attempt at error correction, such as ours, had to suffer the consequences. There was also a high incidence of reverse question construction, such as *Colin Powell is most famous for what?*(835), as well as a heavy reliance on synonym-based back-formation, especially among the last 193 questions. We were, of course, given ample warning that these features would be present, and made some effort to construct regular expressions for the decomposition of reverse-syntax questions.

4.2 Discussion

Our results demonstrate the viability of goal-augmented IR as a preliminary phase in QA, with several important qualifications. First, given that the brute-force IR portion relies on collocation alone, this approach will provide good answers only for questions which provide good clues. Second, the goal identification and extraction mechanism is much better at excluding worthless windows than at seeking out promising ones. In the most pragmatic terms, this means that we can expect fairly high recall (80 to 90% for the top 100 long answers) but fairly low precision (30 to 40% at the first rank). Further and deeper NLP seems to be in order.

It appears that the engineering problems of QA, concerning such things as corpus management, text cleaning, and retrieval of fixed-size passages on the basis of keywords and keyphrases, are largely resolved in our implementation, and that what remains is the fundamental question of how an automatic process can be made to recognize the correspondence between a question (*Who was President Cleveland's wife?*(11)) and its answer ("... Grover Cleveland married Frances Folsom..."). Questions, due to their limited syntactical complexity, are relatively easy to deal with. The most difficult problem in QA remains the analysis of source text, and a good way to handle it seems to be the goal-driven approach, which isolated small, answer-rich snippets of text rather than attacking the whole.

4.3 Future Work

We suggest that a high-performance QA system can be built from a more fine-grained scheme of goal identification and extraction than our current, rather rudimentary one which has achieved a degree of success commensurate with its small scope. It would be best to begin with a subcategorization of proper nouns into such sharply delimited concepts as tall buildings, popular brand names, and famous composers. This would, in effect, be a taxonomy of named entities conceived for the purpose of QA. The extraction of such highly-specialized named entities requires that we have on hand a collection of exhaustive word lists. We propose to compile a wholly new collection by an automatic process performed on a large conventional dictionary.

In addition, we are excited by the prospect of broadening the reach of QA — and perhaps increasing its precision — by opening the floodgates to the Internet. A simple HTTP interface to an online search engine would permit the same sort of information-retrieval processes which our system currently applies on the TREC corpus to be used on documents drawn from the world over. It remains to be seen whether the Web yields useful text passages, but it would be interesting to experiment with the use of various sources, including the "hardwired" corpus, various online works of reference, and the Internet at large, in order to see whether answer confidence may be increased through the correlation of multiple occurrences.

5 Conclusion

Our experiments show that a textual pattern-matching mechanism, properly engineered, can produce satisfactory results on verbose questions of the kind found in the TREC-8 test set. The appeal of such an approach is further amplified when we consider its ease of implementation and modest computational demands. The reason for its success is transparent: in order to elicit a very specific answer, users are often compelled to provide a context-rich question. We have demonstrated the viability of collocation in a full-fledged QA system when it is bolstered by the specialized processes of goal identification and goal extraction. These processes, furthermore, are prerequisite to the extraction of exact-answer strings. It is equally clear that pattern matching is less effective in dealing with terse questions and with those which ask for a definition. In our view, the increasing heterogeneity of questions will have to be met by increasingly specialized QA methods. In particular, we conjecture that definition-based questions are best answered from works of reference, and that one of the vital ingredients in general-purpose QA is an array of tightly constrained goal categories.

References

[ML99] Joel Martin and Chris Lankester. Ask Me Tomorrow: The University of Ottawa Question Answering System. In *Proceedings of TREC-8*, pages 575–583, Gaithersburg, Maryland, 1999.