Halfway To Question Answering

W. A. Woods   Stephen Green   Paul Martin
Ann Houston
Sun Microsystems Laboratories
1 Network Drive
Burlington, MA 01803
{William.Woods,Stephen.Green,Paul.Martin,Ann.Houston}@east.sun.com

1 Introduction

The conceptual indexing and retrieval system at Sun Microsystems Laboratories (Woods et al., 2000) is designed to help people find specific answers to specific questions in unrestricted text. It uses a combination of syntactic, semantic, and morphological knowledge, together with taxonomic subsumption techniques, to address differences in terminology between a user's queries and the material that may answer them. At indexing time, the system builds a conceptual taxonomy of all the words and phrases in the indexed material. This taxonomy is based on the morphological structure of words, the syntactic structure of phrases, and semantic relations between meanings of words that it knows in its lexicon. At query time, the system automatically retrieves any concepts that are subsumed by (i.e., are more specific than) the user's query terms, according to this taxonomy.

The system uses a penalty-based relaxation-ranking method to locate and rank potentially relevant passages in the material. It provides a user with feedback on why passages were retrieved, so that irrelevant passages can be quickly skipped over, and it provides information about where query terms fit in its conceptual taxonomy, so that users can see opportunities for generalizing queries if the first request is not successful or more information is desired. This methodology, which falls somewhere between traditional document retrieval and TREC-style question answering, has proven to be very effective in reducing the time required for people to find information in online material (Woods et al., 2000).

For the most part, the conceptual indexing project has focussed on finding techniques for improving human search productivity, rather than dealing with the problems of large collections. However, one experimental version of this system, which we internally call Nova, has recently reached the point where it can index collections the size of the TREC corpora. In addition to Nova, we have a pilot system that includes much stronger morphology and phrase extraction components and a newer lexicon; however, the pilot system is currently limited in the amount of text that it can process.

We decided to enter a system based on Nova in the question-answering track of this year's TREC competition in order to see how far we would have to go to transform Nova into a full question-answering system. From our experience with Nova, it seemed that it might provide a good first-stage retrieval for a question-answering system because it is able to find specific passages where the terms of a request occur near each other and can automatically deal with some of the paraphrase differences between the wording of a request and the wording of a relevant passage. Nova is able to find good passages directly from the conceptual index, without first performing a separate document retrieval step.

We entered the TREC competition very late in the game, so we had only a couple of weeks to download and index the TREC collections, implement some quick extensions to Nova, and run an experiment. We indexed the TREC material using an option in Nova that indexes all word occurrences and their subsumers. We had no previous exposure to any of the TREC collections, so Nova's lexicon and conceptual taxonomy have had no previous adaptation to this material.

We discovered that with the addition of a simple automatic query formulation algorithm, Nova performs modestly well at the 250-byte question answering task, finding a correct answer in the top 5 hits roughly half of the time. The remaining cases tend to fall into three categories:

1. Cases where a key passage was not found due to a paraphrase relationship that was not yet covered in our conceptual taxonomy. For example, "The first Russian astronaut to spacewalk" was described as a "Soviet cosmonaut" in a desired passage, and our taxonomy doesn't yet know that Soviet means Russian or that cosmonaut is a kind of astronaut. If these facts were in our lexicon, then Nova would have gotten the correct answer at rank 1.

2. Cases where glitches in the current version of Nova or our experimental lashup (which combined Nova and parts of the pilot system) caused it to fail to find things that it should have. See the failure analysis in section 5.2 below for examples.

3. Cases where nonessential terms in the query could not be found in the desired passages, but could be found elsewhere in the material in combination with almost all of the other elements of the query, but missing an essential element. For example, in question 411, "What tourist attractions are there in Reims?", we found numerous tourist attractions elsewhere, but none of the passages that mention Reims contains either *tourist* or *attraction* or anything subsumed by them in our taxonomy.

We also discovered that the Nova system itself is a highly useful tool for investigating why a given passage was not retrieved. With a Nova index to the entire collection, we can quickly find passages containing combinations of specified terms, and can use general terms in the request that may subsume more specific terms in the text. We can view these passages in decreasing order of quality, and jump quickly to the corresponding positions in the source documents. We can also quickly survey how given terms are used in the collection and how they relate to other terms in our taxonomy. For example, it took only a few tries to discover the "Soviet cosmonaut" variation of the above-mentioned request.

2 Getting from Nova to Question Answering

As mentioned previously, Nova does something between document retrieval and question answering: It identifies useful places in the documents for a human searcher to look, and it provides information to help a person make quick assessments about whether a passage is likely to be relevant, but it doesn't really understand either the question or the answer. The TREC QA task requires more than that.

Nova is missing two key components necessary to perform the TREC question-answering task:

1. A query-formulation component to determine the answer type of the desired result and to transform the natural language question into a more effective query.

2. An answer-location component for locating answers in or near the retrieved passages.

For this competition, we provided Nova with simple baseline versions of these two components.

The query-formulation component is based on our pilot system. The query formulator interprets the question words and the format of the question to determine the desired answer type. It also either replaces the question word in the query with the desired answer type or simply removes it from the request.

The answer-location component determines a 250-byte passage of text that may contain the answer to the question. Nova returns passages that typically range from a single term to a passage the size of a paragraph or more and also returns approximately 200 bytes of context on either side of the passage. We have observed that the answer to a Nova query is often either in the retrieved passage or the neighboring context. The task of the answer-location component is to select one or two 250-byte passages from the (typically 500 to 1000 byte) passage returned by Nova.

This approach differs from most systems in the QA track in that we used Nova to index and search the entire TREC 9 collection, eliminating the stage of using a preliminary document retrieval system to select a subset of the collection for further processing.

3 Run Details

3.1 Query formulation

Questions have a logical structure consisting of an answer type and a relational condition. Answering a question consists in finding an instance of the answer type that satisfies the relational condition. Like other systems, we implemented a function to interpret a question into such a structure. For example, for one of the TREC 8 questions, "What was the monetary value of the Nobel Peace Prize in 1989?", this function produced the structure in figure 1. This structure indicates that a number is sought as the answer and that `MONETARY VALUE OF NOBEL PEACE PRIZE IN 1989` is to be used as the Nova query.

```
(NUMBER (MONETARY VALUE OF NOBEL PEACE PRIZE IN 1989))
```

Figure 1: Question structure produced by query formulation.

The query-formulation stage determines the answer type, replaces the question word with the answer type (or sometimes just removes it), and often changes the wording of the relational condition in one or more of the following ways:

- Shorten it if necessary, by dropping less important terms. This step is necessary because Nova's design point has been short questions and the current implementation does not use more than ten words of a request. Previously, the longest query we had encountered was eight words, and most requests consisted of three or fewer words.

- Generalize some terms (e.g., *high* for *tall*), substitute base forms for inflected forms (e.g., *principle* for *principles*), and drop some "noise" terms.

Like other TREC QA systems, for example, (Moldovan et al., 2000; Breck et al., 2000), this first stage consists of a dispatch table based on the question words and their pattern of use in the question. The patterns in Table 1 roughly characterize the rules for determining the answer type.

Question word	Answer type
whose	person
(who whom)	person
(which what)	depends on subsequent words
prep (which what whom)	do case analysis
when	date
where	location
why	reason
how (many much long)	number
how *adjective*	number
how	way
name	name
otherwise	look for embedded question words

Table 1: Dispatch table for question words.

These rules are similar to those used in other TREC systems. In addition, we performed a number of other transformations on the rest of the question:

- Combine elements like *U.S* from *U . S* and strip out other punctuation.
- Delete elements like *name of*, *most of*, *day of week*, *time of day*, *can be*, *may be*, *can not*, *must not*, *have been*, *take place*, *held at*, *does it take*.
- Replace inflected forms of words with their base forms.
- Generalize certain words (e.g., *tall* to *high*).
- If query is too long, skip parenthetical comments in parentheses, unless they are names.
- If query is still too long, delete words from a stop word list.
- If query is still too long, delete less important elements from a priority-ordered list until query is short enough or it runs out of things to delete (in which case simply truncate it to 10 words).

These rules were put together quickly based on examining some of the TREC 8 questions.

3.2 Answer location

For the answer-location component, we tried two simple approaches, embodied in the runs we labeled SunOne and SunToo, both of which used the query-formulation component described above. For SunOne, we used the pilot system to locate all occurrences of the desired answer type that occurred in Nova's retrieved passage or its neighboring context and then selected one or two 250-byte passages to include as many candidate answers as possible. No effort was made to determine the actual relationship between these candidates and the content of the query. For SunToo we simply truncated passages longer than 250 bytes to the leftmost 250 bytes, and symmetrically extended shorter passages to include a full 250 bytes.

We thought these approaches might give a useful approximate answer location because we had noticed that Nova often gets the desired information as the first hit[1], either in the passage itself or in the neighboring context region that Nova returns with the passage.

There are a number of different situations in which one or more instances of the desired answer type can occur (or not occur) in or near the retrieved passages, so the following rules were used in the SunOne case to select a 250 byte passage in the various cases:

1. No answer type in or near passage and passage is not longer than 250 bytes. Extend both ends of the passage.
2. No answer type in or near passage and passage is longer than 250 bytes. Shrink the passage by cutting off the right end.
3. Answer types at both ends but all found answer types fit within 250 bytes. Extend both ends of the passage to include all answer types.
4. Found answer types span more than 250 bytes, so split the passage into two pieces (the left and right halves) and slightly favor one (usually the left) in the ranking.
5. Answer types at left end only. Shift passage to the left.
6. Answer types at right end only. Shift passage to the right.

These rules indicate how to move the ends of the passages that Nova returns in order to determine the 250 bytes to be used. In case number 4, instances of the answer type were found sufficiently far apart that the Nova passage was split into two and both were generated as candidates, with the leftmost being favored in most cases, and the rightmost in some cases.

3.3 Results

The results of the two runs are shown in Table 2, where the scores are labeled to indicate the strict and lenient human judgements and the scores that were derived for the training set from the automatic eval script provided for TREC 8 as extended by us to include some answers we found in the TREC 9 material that were not covered in the script. Interestingly, although the SunOne strategy outperformed the SunToo strategy in our testing on the TREC 8 questions, they perform almost equally on the TREC 9 test, with the simple SunToo case slightly outperforming SunOne. For both strategies, the TREC 9 questions were substantially more difficult than the TREC 8 questions.

After the formal submission, we discovered a low-level paging bug in the Nova retrieval system, the effect of which was to make a portion of Nova's conceptual taxonomy invisible to the search process. This resulted in noticeable errors in the system performance such as sometimes failing to find a hit on an inflected form of a word when the query term was a base form of that word, a capability that Nova generally does quite well. After the formal submission and after receiving the eval script that allows us to automatically score a run, we reran the system with this bug fixed but with no other changes. We rescored both the submitted run and the run with the bug fix, using the eval script, to determine the effect of the bug and to calibrate the difference between the eval script scores and the human judgement scores.

[1]Getting the correct information near the top and making it easy for a person to skip over irrelevant passages has been the main goal; getting it to actually be first was considered a bonus.

The results are shown in Table 3. In this table, we estimated the equivalent strict human judgements for the system with the bug fixed by assuming that the ratio of the eval script MRR scores to the human judgement MRR scores is a constant and similarly for the success rate scores. Notice that although the result is a 10% improvement in MRR and 12% improvement in success rate, this improvement is the net result of improving on some questions and losing the answers (from the top 5) or moving down in the ranking for some others. Because the nature of the bug was to hide a portion of the taxonomy, this turns out to be an inadvertent experiment that reconfirms the results of several previous experiments which show that increasing the amount of knowledge in the taxonomy results in an improvement in the retrieval results for this technology.

	Mean reciprocal rank	Success rate at 5 hits
training set:	TREC 8 questions	(on TREC 9 data)
SunOne	0.50 (eval script)	64.6% (eval script) (71.2% at 10 hits)
SunToo	0.44 (eval script)	55.1% (eval script)
submitted:	TREC 9 questions	(on TREC 9 data)
SunOne	0.340 (strict) 0.348 (lenient)	46.2% (strict) 47.4% (lenient)
SunToo	0.345 (strict) 0.354 (lenient)	46.9% (strict) 47.5% (lenient)

Table 2: Results of two Sun runs on TREC 8 and TREC 9 questions.

	Mean reciprocal rank	Success rate at 5 hits
submitted:	TREC 9 questions	(with paging bug)
SunToo	0.345 (strict) 0.369 (eval script)	46.9% (strict) 48.39% (eval script)
corrected:	TREC 9 questions	(with bug fixed)
SunToo	0.380 (strict*) 0.406 (eval script)	52.6% (strict*) 54.25% (eval script)
improvement:	10%	12%
net gain:	picked up 73 new answers	and lost 33 old ones; net gain 40

Table 3: Results of correcting the paging bug. (* indicates estimated strict results)

4 Question Analysis

Because our results depend in part on our ability to formulate good queries from the questions, and this may depend on the type of the question, it is informative to look at the distribution of question types in the TREC queries and our performance on the different types. Table 4 shows the number (and percentage) of different question types from both TREC 8 and TREC 9 and our corresponding performance.

Note that two of the three question types for which our success rates were the lowest (*how* and *what*) are those which have the greatest diversity of entries in Table 1, and are cases where our analysis to identify the answer type for the question is somewhat limited. We also note that there were a variety of new kinds of *what* questions in TREC 9 that did not occur in TREC 8 and more than half of the TREC 9 questions are *what* questions. This partly explains why TREC 9 is a substantially more difficult task.

Question type	TREC 8		TREC 9		TREC 9 results				
how	31	(15.9%)	54	(8.4%)	SunOne:	MRR:	0.122	SR:	25.9%
					SunToo:	MRR:	0.162	SR:	29.6%
what	65	(33.3%)	374	(58.3%)	SunOne:	MRR:	0.315	SR:	43.3%
					SunToo:	MRR:	0.322	SR:	43.3%
when	19	(9.7%)	49	(7.6%)	SunOne:	MRR:	0.290	SR:	42.9%
					SunToo:	MRR:	0.319	SR:	46.9%
where	21	(10.8%)	72	(11.2%)	SunOne:	MRR:	0.398	SR:	54.2%
					SunToo:	MRR:	0.447	SR:	59.7%
which	10	(5.1%)	13	(2.0%)	SunOne:	MRR:	0.359	SR:	46.2%
					SunToo:	MRR:	0.308	SR:	46.2%
who	47	(24.1%)	117	(18.3%)	SunOne:	MRR:	0.500	SR:	60.7%
					SunToo:	MRR:	0.450	SR:	58.1%
why	2	(1.0%)	3	(0.5%)	SunOne:	MRR:	0.444	SR:	66.7%
					SunToo:	MRR:	0.500	SR:	66.7%
other	5	(2.5%)	0	(0.0%)					

Table 4: Distribution of question types for TREC 8 and TREC 9.

5 Results Analysis

In order to try to understand the results of this experiment, we have begun to look in some detail at the behavior of the experimental lashup. So far we have done three analyses: a random sample study of 10 randomly chosen questions, a failure analysis of a selected set of questions on which we did less well than other systems, and a comparison of the results of different paraphrases.

For the random sample, it turns out that all of the cases where no correct answer was found are cases where Nova failed to find at least one of the requested terms in the best ranked passage. This is not true in general, but it is a strongly correlated indicator. Table 5 shows the mean reciprocal rank and the success rates broken down by whether the first hit has no missing terms or the first hit has at least one missing term. This data suggests that one way to use Nova for question answering would be to reformulate the question and ask again when there is a missing term in the best hit. This is typical of the way that humans use Nova.

	Number of questions	Mean reciprocal rank	Success rate
SunOne: No missing term	444	0.417 (strict)	55.9%
SunOne: Missing term	238	0.196 (strict)	28.2%
SunToo: No missing term	444	0.435 (strict)	57.2%
SunToo: Missing term	238	0.177 (strict)	27.7%

Table 5: Performance with respect to missing terms in best hit.

Like other systems in TREC 8, we have observed a bimodal distribution of results — we tend to either do fairly well on a question or we miss it entirely.

5.1 Random sample analysis

Table 6 shows the results for 10 randomly chosen questions from the TREC 9 test set (using the strict SunOne results, and showing the answer type and Nova query that were used).

Num	Question	Nova query	Result
402	What nationality was Jackson Pollock?	(NATIONALITY (NATIONALITY JACKSON POLLOCK))	Rank 1
455	What is Colin Powell best known for?	(THING (COLIN POWELL BEST KNOW FOR))	Rank 5
459	When was John D. Rockefeller born?	(DATE (JOHN D ROCKEFELLER BORN))	Rank 2
487	What was the name of the movie that starred Sharon Stone and Arnold Schwarzenegger?	(THING (MOVIE STAR SHARON STONE ARNOLD SCHWARZENEGGER))	Missing “Sharon” in first hit
576	What is the name of Joan Jett’s band?	(THING (JOAN JETT BAND))	Rank 3
616	What is the purpose of a car bra?	(THING (PURPOSE OF CAR BRA))	Missing “bra” in first hit
657	In what country is a stuck-out tongue a friendly greeting?	(COUNTRY (IN COUNTRY STUCK-OUT TONGUE FRIENDLY GREET))	Missing “stuck-out” and “tongue” in first hit
711	What are the names of the tourist attractions in Reims?	(THING (NAMES OF TOURIST ATTRACTION IN REIMS))	Missing “Reims” in first hit
803	What king signed the Magna Carta?	(THING (KING SIGN MAGNA CARTA))	Rank 1
865	What is the meaning of caliente (in English)?	(THING (MEAN OF CALIENTE IN ENGLISH))	Missing “caliente” in first hit

Table 6: Random sample of questions and results.

The questions in this sample for which a correct answer was not found, all of which (as mentioned) missed at least one term in the best passage Nova could find, were investigated using the Nova system interactively to try to find passages that would answer the question. The results are as follows:

For question 487, “What was the name of the movie that starred Sharon Stone and Arnold Schwarzenegger?,” a correct passage is found at rank 1 if “movie” is replaced by “film”, even though it is missing a term for “star”. Our pilot system taxonomy knows that “movie” is equivalent to a sense of “film”, but the taxonomy used by Nova doesn’t know this yet.

For question 616, “What is the purpose of a car bra?,” the only thing we could find was a “stealth car bra” which fools police radar (rank 3 for “car bra” and repeated multiple times in the collection). There were no other occurrences of car bra in the collection, except for one consumer complaint about having purchased one. Apparently the human judges considered the “stealth car bra” correct, but this is not the purpose of an ordinary car bra.

For question 657, “In what country is a stuck-out tongue a friendly greeting?,” the only instances of stuck-out tongue were for Mr. Yuk, a children’s symbol for poison. No entrant got a correct answer for this question.

For question 711, “What are the names of the tourist attractions in Reims?,” the submitted query included the “names of”, which favored hits that mentioned “names” at the expense of “Reims”. Dropping that and simply asking “tourist attraction in Reims” still was missing Reims. Trying “sight in Reims” didn’t do it either. Simply asking for Reims gets Reims Cathedral in the third hit.

For question 865, “What is the meaning of caliente (in English)?,” the query “caliente” gets a correct answer at rank 10.

5.2 Failure analysis

We looked in some detail at cases where other systems seem to have done better than Nova. One strong pattern that emerged is that a number of glitches of various sorts blocked many passages from being retrieved that would otherwise have been handled by this methodology. Many questions failed due to bugs and/or missing facts in our lexicon, and the fact that we used a newer version of the lexicon and morphology for the query-formulation and answer-location parts of the system than was used in the version of Nova that we were using to find the passages.

For example, in question 515, the word *Sinemet*, which was not in our lexicon, was aggressively analyzed morphologically (Woods, 2000) as the past tense of *sinemeet* and the query-formulation component substituted *sinemeet* for *sinemet* in the query given to Nova. Although wrong, this would not have caused a problem if the same version of the morphology had been used in Nova, since *Sinemeet* would have subsumed *Sinemet* and retrieved the passage correctly. (The morphological component has since been fixed to avoid such analyses of words ending in *met*.) With this fix, the correct passage is found at rank 2.

Another pattern that emerged is that Nova is currently very generous in its subsumption analysis, allowing a subsumption if any sense of a word is subsumed, without any attempt to disambiguate the sense of the word in the text. Since Nova is also knowledgeable about many names that are also ordinary words in English (like *bill* and *woods* and *green*), many of which are also city names, it finds a number of subsumptions of ordinary words under *person* and *place*. This sometimes interacts badly with the question-answering strategy.

Finally, as described above, in the version of Nova that we ran, there was a low-level paging bug that effectively hid part of the conceptual taxonomy and, among other things, failed to retrieve inflected forms of some words in some cases (e.g., *hexagons* was not retrieved in response to a query including *hexagon*). We have identified a number of queries where this bug kept us from finding answers, and the experiment described in Table 3 shows that fixing this single bug results in a 10-12% improvement.

5.3 Sensitivity to paraphrase variations

To assess the sensitivity of this methodology to paraphrase variations, we compared the results for different questions in the TREC 9 paraphrase sets. Table 7 gives an overview of how well the system handled different paraphrases on a question-by-question basis. This table shows the question numbers of questions in the TREC 9 paraphrase sets, followed by the respective rank scores at which SunToo found answers (the NILs are for questions for which there was no result reported in the TREC 9 results). From these numbers you can see that SunToo did consistently well or consistently poorly on many paraphrases, e.g.:

(450 863 864 865 866) (0 0 0 0 0)

(454 830 831 832 833 834) (1 1 1 1 1 1)

However, there are other cases in which the details of the paraphrase made a big difference, e.g.:
(394 808 809 810) (1 0 0 1)
394 What is the longest word in the English language?
808 What English word has the most letters?
809 What English word contains the most letters?
810 What is the longest English word?

Paraphrase set	Reciprocal ranks	Paraphrase set	Reciprocal ranks
(201 732 733 734)	(0 0 0 0)	(203 728 729 730 731)	(0 0 0 1/4 1/2)
(393 805 806 807)	(1 1 0 0)	(394 808 809 810)	(1 0 0 1)
(396 811 812 813)	(0 NIL 0 0)	(397 814)	(0 0)
(398 815 816 817)	(0 0 0 0)	(400 867 868 869)	(1 0 0 0)
(402 870 871 872)	(1 1 1 1)	(403 873 874 875 876 877)	(0 0 0 1 0 0)
(404 878 879 880 881 882)	(0 0 0 0 0 0)	(405 883 884 885 886 887)	(1/4 0 1 1 0 1)
(406 888 889 890 891)	(0 1/2 1/3 1/4 1/2)	(407 892 893)	(1/5 1 1/2)
(408 701 702 703 704)	(1/2 1/2 0 1 1/2)	(409 705 706 707 708)	(1 1/2 0 0 1/2)
(410 709 710)	(0 0 0)	(411 711 712 713 714 715 716 717)	(0 0 0 0 0 0 0 0)
(412 718 719 720 721 722)	(0 0 0 0 0 0)	(413 723 724 725 726 727)	(1 1 1/2 1/5 0 1/2)
(415 735 736)	(0 1 1/2)	(416 737 738 739 740 741)	(0 0 0 0 0 1)
(417 742 743)	(1 1 1)	(418 744 745)	(1 1 1)
(419 746 747 748 749)	(1 1 0 1 1)	(421 750 751 752)	(1 0 0 0)
(423 753 754 755 756 757)	(0 0 0 0 0 0)	(424 758 759 760 761)	(0 0 0 0 0)
(425 762 763 764 765)	(1/2 0 1 0 1/2)	(426 766 767 768)	(1/2 0 1/3 1/2)
(427 769 770 771)	(0 0 0 0)	(428 772 773 774 775 776 777)	(1 1 1/3 1 1 1 1)
(429 778 779 780)	(1/3 1/3 1/3 1)	(431 781 782 783 784)	(0 0 0 0 1)
(433 785 786)	(0 0 0)	(435 787 788 789)	(1/2 0 0 0)
(436 790 791 792 793)	(0 1/3 0 1/3 0)	(437 794 795 796)	(0 NIL 0 NIL)
(440 797 798 799)	(1 0 0 0)	(441 800 801 802 803 804)	(1/2 1 1 1/2 1 1)
(442 844 845 846)	(1 0 1/5 1)	(444 847 848 849 850)	(1/5 1 1 1 1/2)
(445 851 852 853)	(0 1/5 0 1)	(446 854 855)	(1/5 1/2 1/2)
(448 856 857 858 859)	(0 1 0 0 1)	(449 860 861 862)	(1 0 1 1)
(450 863 864 865 866)	(0 0 0 0 0)	(451 818 819 820 821 822)	(1 0 0 0 0 0)
(452 823 824 825 826 827)	(1 0 1 1 1 1)	(453 828 829)	(1/3 1 1)
(454 830 831 832 833 834)	(1 1 1 1 1 1)	(455 835 836 837 838)	(0 1/4 0 1 1/2)
(456 839 840 841 842)	(1 0 0 1/2 1)	(458 843)	(0 1)

Table 7: Paraphrase sets and respective results.

A detailed analysis of these cases reveals the kinds of paraphrase variations that the system does not yet handle. In the above example, the system doesn't have a way to relate “the longest word” to “the word containing the most letters”.

6 Query Track Experiment

Although we did not intend to participate in the TREC Query Track, and Nova is not a document retrieval system, we responded to the NIST request to run our system on the query track queries. We did this by modifying Nova to simply return the document id without the passage-specific information and to give each document the score and rank of its best passage. Previous experiments comparing this kind of modified passage retrieval to more traditional document retrieval systems had shown that the passage-retrieval-based system found relevant documents that the traditional system did not, and vice-versa[2]. The former tends to find documents that contain on-point passages even when the document overall may be about something else, while the latter tends to find documents that make repeated references to the terms in the query. For example, in the man-page experiment in (Woods et al., 2000), the system found the File Manager man page as an answer to "print a file" because of a paragraph explaining the function of the "print" button in the file manager, but the human judge that had previously determined the set of relevant documents hadn't thought of the File Manager man page as a relevant document.

We tried this modified Nova system in two versions: one (SUN) that took the query as given and passed it to Nova, and another (Sunl) that first applied the query-formulation procedure described in section 3.1 and passed the result to Nova. The latter performed better on this task because the former effectively truncates queries beyond the first 10 words and many of the queries have more than 10 words.

Since Nova was designed for specific information requests that need only one answer, the goal has been to get the best relevant passage in the first ten hits. Multiple hits for the same question have not been important. Consequently, we sacrifice high recall for the ability to drill in on precise content, and we don't score well on the MAP measure because of its dependence on recall. The figure of merit that interests us is the success rate, and that is what we measured in this experiment.

	Success rate at					
	20 docs		10 docs		5 docs	
Average for	Sunl	SUN	Sunl	SUN	Sunl	SUN
All Sets	74.56%	66.00%	64.70%	56.65%	54.37%	46.65%
Verbose Sets	70.83%	56.00%	60.50%	46.33%	50.33%	36.50%
Concise Sets	79.26%	78.63%	70.00%	69.68%	59.47%	59.47%

Table 8: Success rates for verbose and concise query sets.

The most salient difference between the query sets is the difference between the sentence cases, which have relatively long questions, complete with question words (e.g., Question 94 in set INQ2a "What kind of illegal activities can be performed with the aid of computers"), and the rest, which have relatively short topic descriptions (e.g., Question 94 in set INQ1j "Illegal computer activity"). In general, the Nova system works better on the concise topics, and more specifically on concise topics that relate to specific information. Table 8 gives the success rates for the two Query Track runs and the breakdown for verbose and concise query sets.

[2]Given this fact, it should be noted that because the relevance judgments for this experiment are based on previous TREC results, many of the "non-relevant" documents that we retrieved may actually be relevant, but were not found in the previous TREC results. Indeed, we have found a number of cases where we retrieved documents that were not judged relevant but appeared relevant to us.

7 Conclusions

Although the Nova system is not a question answering system, it is an interesting alternative for the first-phase retrieval that precedes more detailed linguistic processing in most question answering approaches. An experimental lashup using Nova with a simple query-formulation algorithm gets the correct answer in the top 5 choices approximately half the time on the TREC 9 task, without any real understanding of either the questions or the passages that it retrieves and using general-purpose linguistic knowledge that has had no previous exposure to the TREC material. The system could be improved greatly by expanding its knowledge and eliminating bugs in the experimental lashup. However, to achieve a real question-answering capability, we clearly need to move beyond the simple baseline strategies used here for query formulation and answer location.

In addition, the Nova system is a useful tool for analyzing the results of these experiments by helping a researcher find passages that could answer a question, so that the reasons for failure can be understood.

References

(Breck et al., 2000) Eric Breck, John Burger, Lisa Ferro, David House, Marc Light, and Inderjeet Mani. A Sys Called Qanda. In *Proceedings of The Eighth Text REtrieval Conference (TREC 8)*. National Institute of Standards and Technology, 2000.

(Moldovan et al., 2000) Dan Moldovan, Sanda Harabagiu, Marius Paşca, Rada Mihalcea, Richard Goodrum, Roxana Girju, and Vasile Rus. LASSO: A Tool for Surfing the Answer Net. In *Proceedings of The Eighth Text REtrieval Conference (TREC 8)*. National Institute of Standards and Technology, 2000.

(Singhal et al., 2000) Amit Singhal, Steve Abney, Michiel Bacchiani, Michael Collins, Don Hindle, and Fernando Pereira. AT&T at TREC-8. In *Proceedings of The Eighth Text REtrieval Conference (TREC 8)*. National Institute of Standards and Technology, 2000.

(Srihari and Li, 2000) Rohini Srihari and Wei Li. Information Extraction Supported Question Answering. In *Proceedings of The Eighth Text REtrieval Conference (TREC 8)*. National Institute of Standards and Technology, 2000.

(Voorhees and Harman, 2000) E.M. Voorhees and D.K. Harman, editors. *Proceedings of The Eighth Text REtrieval Conference (TREC 8)*. National Institute of Standards and Technology, 2000.

(Woods et al., 2000) William A. Woods, Lawrence A. Bookman, Ann Houston, Robert J. Kuhns, Paul Martin, and Stephen Green. Linguistic Knowledge can Improve Information Retrieval. In *Sixth Annual Applied Natural Language Processing Conference*, pages 262–267. Association for Computational Linguistics, 2000.

(Woods, 2000) William A. Woods. Aggressive Morphology for Robust Lexical Coverage. In *Sixth Annual Applied Natural Language Processing Conference*, pages 218–223. Association for Computational Linguistics, 2000.