Question Answering Using a Large Text Database: A Machine Learning Approach

Hwee Tou Ng*
Jennifer Lai Pheng Kwan
Yiyuan Xia
DSO National Laboratories
20 Science Park Drive
Singapore 118230
{nhweetou, klaiphen, xyiyuan}@dso.org.sg

Abstract

In this paper, we present a machine learning approach to question answering. The task is answering factual questions, where the answers are to be found in documents in a large text database. We trained our system on 398 questions from the Remedia corpus, as well as 38 TREC-8 development questions. We then evaluated our system on 198 questions of the TREC-8 question answering task. Although our learning approach only uses 4 features, we are able to achieve quite competitive accuracy. The results indicate that such a machine learning approach is a promising way to build a state-of-the-art question answering system.

1 Introduction

Finding information in a large text database is the traditional subject of study in information retrieval. When given some keywords, current search engines retrieve numerous web pages that contain the keywords, and leave it to a user to sieve through the large set of returned web pages to find the information that he needs. That is, search engines deal more with whole document retrieval.

However, what a user often wants is really a precise answer to a question. For instance, given the question "When was the last US presidential election held?", what he wants is the answer "Nov 7, 2000", rather than to read through lots of web pages that contain the words "US", "presidential", "election", etc to find the date of election. That is, what a user needs is *information* retrieval, rather than the current *document* retrieval.

Question answering (QA) has recently attracted a lot of research activities. This is in part fueled by the question answering track of TREC. TREC is an annual exercise to evaluate the performance of text retrieval systems on common, large real-world text collections, using a uniform scoring procedure. The question answering track started in TREC-8 in 1999 (Voorhees, 2000; Voorhees and Tice, 2000, TREC). In the question answering track, for each factual question, the task is to extract the top five 50-byte or 250-byte answers to the question from a large text database consisting of hundreds of thousands of documents (gigabytes of text).

Another strand of work that deals with question answering surfaces in the context of reading comprehension. The group at MITRE (Hirschman et al., 1999) initiated work on reading comprehension of children stories. In particular, the task involves choosing a sentence from a story that best answers a question posed to the story. For this task, there is no need to deal with retrieval from a large text database, since a question is directed at a particular story.

In this paper, we address the task of answering factual questions, where the answers are to be found in documents in a large text database. We adopt a machine learning approach to question answering. In particular, answer candidates are classified and ranked by a classifier trained on a set of question-answer pairs.

2 A Machine Learning Approach

2.1 Natural Language Processing Modules

Our question answering system utilizes a number of natural language processing (NLP) modules. They include sentence segmentation, tokenization, morphological analysis, part-of-speech tagging, noun phrase chunking, named entity tagging, and semantic class determination. The main goal of these modules is to identify the boundary and the semantic class of the noun phrases, so that the necessary feature values needed to form the training or test examples can be computed.

Our part-of-speech tagger is a standard statistical bigram tagger based on the Hidden Markov Model (HMM) (Church, 1988). Similarly, we built a statistical HMM-based noun phrase chunker where the noun phrase boundaries are determined solely based on the part-of-speech tags assigned to the words in a sentence. We also implemented a module that assigns named entity tags. In particular, the following named entity types are recognized: *human, orga-*

* Hwee Tou Ng is also affiliated with Department of Computer Science, School of Computing, National University of Singapore, http://www.comp.nus.edu.sg/~nght

nization, location, date, time, percent, and *money.* Our named entity tagger uses the HMM approach of (Bikel et al., 1999), which learns from a tagged corpus of named entities. Our part-of-speech tagger, noun phrase chunker, and named entity tagger achieve state-of-the-art accuracy.

The semantic classes defined for noun phrases in our QA system are: *human, organization, location, date, time, percent, money,* and *entity.* Each of the 7 semantic classes *human, organization, ..., money* is a subclass of the semantic class *entity,* which is a catch-all semantic class for all noun phrases that are not of the other 7 defined semantic classes. Each of these semantic classes is then mapped to a WORDNET synset (Miller, 1990). Our semantic class determination module assumes that the semantic class for every noun phrase extracted follows the first sense of the head noun of the noun phrase. Since WORDNET orders the senses of a noun by their frequency, this is equivalent to choosing the most frequent sense, which then determines the semantic class of the noun phrase through upward traversal of the ISA hierarchy of WORDNET.

2.2 Features

The learning task is defined as identifying the best noun phrase that is most likely to answer a given question.

Our feature vector consists of 4 features. Each feature vector is derived from one triple q, n, and s, where q is the question, n is the noun phrase, and s is the sentence containing the noun phrase n.

- Question type (QT)

  This feature attempts to capture the focus of a question q, i.e., what a question is asking for. Its possible values are *who, when, where, how, human, organization, location, date, time, percent, money,* and *entity.*

  The value for this feature is determined as follows. Step through each successive word w in q (starting from the first word in q), and check if w is one of the words "who", "whom", "whose", "when", "where" and "how" (in that order). The first match found will determine the question type of q. If the first matching word is one of "who", "whom", or "whose", then the question type is *who.* If the first matching word is "when", the question type is *when.* Similarly for *where* and *how.*

  If no match is found, the following heuristic is used to determine the question type. The heuristic searches for the first noun phrase in the question that does not occur after a non-"be" verb, and whose head word is not the word "name". The semantic class of this head word is then used as the question type. For example, in the question "What is the name of the managing director of Apricot Computer?", the first noun phrase satisfying the conditions is "managing director" and the semantic class of its head word "director", which is *human,* is the question type.

  If no such head word can be found, then the question is given the type *entity.*

- Noun phrase semantic class (NPSC)

  The possible values of this feature are *human, organization, location, date, time, percent, money,* and *entity.*

  If a named entity tag is assigned to the noun phrase n, then NPSC is assigned the value of the named entity tag. Otherwise, NPSC is assigned the value of the semantic class of n.

- Quantitative noun phrase (QNP)

  The possible values are true and false.

  If the noun phrase n contains a number and its NPSC is not *date* or *time*, then this value is true. Otherwise, this value is false. This feature is indicative of answers to "how many", "how far", "how long", etc type of "how" questions.

- Diff-from-Max-Word-Match between s and q (DMWM)

  The possible values are 0, 1, 2, 3, ... This feature attempts to capture the word overlap between a question and a sentence. It is similar to that used in our previous work (Ng et al., 2000).

  To compute the value of this feature, all words in the question q and the sentence s (which contains the noun phrase n) are reduced to their morphological roots. Let m be the number of morphological root words in q which can be found in s. Stop words are excluded in the count. Then for all the possible sentences s_i in the defined search space[1], let m_i be the number of morphological root words in q which can be found in s_i. Define $M = max_i\{m_i\}$. The value of this feature is then computed as $M - m$. That is, for a sentence s that has the maximum number of words that overlap with q (among all sentences in the defined search space), its value for this feature will be 0.

2.3 Training

Given a question q, a noun phrase n that is marked as an answer to q, as well as the sentence s containing n, the triple q, n, and s are used to generate a positive training example, as described in Section 2.2. To compute the value of DMWM, the search space

[1] The list of sentences considered in this search space will be explained in the following subsections on training and testing.

is defined as all the sentences in d, where d is the document containing s.

Negative training examples are generated by randomly selecting one noun phrase (other than n) each from sentences s, s_{-1} and s_{+1}, where s_{-1} is the sentence immediately preceding s, and s_{+1} is the sentence immediately following s.

A classifier is then built based on the feature vectors generated from the training questions and answers. The learning algorithm used is C5 with boosting (Freund and Schapire, 1996). C5 is a more recent version of C4.5 (Quinlan, 1993). We use the default values for all C5 learning parameters.

2.4 Testing

Before identifying all possible noun phrases for testing, the search space of all documents in a large text database will have to be efficiently narrowed down first.

2.4.1 Passage retrieval

The retrieval system we use is based on that described in (Singhal et al., 2000). For a given question, query terms (i.e., words in the question) are first extracted by removing stopwords and punctuation symbols. A term weight is then assigned to each query term t, using the *idf* factor: $log(1 + \frac{N}{d_t})$ where N is the total number of documents in the text database, and d_t is the number of documents in the database containing term t.

Within a document, a passage is any contiguous text string which contains up to a maximum of 5 consecutive sentences, or at most 500 bytes. Each candidate passage in a document is assigned a score, based on the sum of the weights of all query terms contained in the passage. See (Singhal et al., 2000) for more details of how passage is selected and scored.

In our experiments with TREC-8 test data, we used the top 200 documents per question provided by AT&T's retrieval engine. This set of documents is also provided to all TREC-8 participants. The 3 top-scoring passages of each of the top 200 documents are selected, and these passages are ranked in descending order of the passage score. The top 200 passages are then passed to the answer ranking module described in Section 2.4.2.

2.4.2 Answer ranking

Each noun phrase in the retrieved passages contributes a test example. The question q, the noun phrase n, and the sentence s containing n are used to generate a test example. For a given question q, the value of M needed in the computation of DMWM is taken over the search space of all the sentences in the retrieved passages for q. The classifier assigns a positive or negative class together with a confidence value to each test example. The noun phrases are then sorted according to their classification and confidence values. Noun phrases with positive class are ranked above noun phrases with negative class. If 2 noun phrases have the same positive class, then the one with the higher confidence score is ranked higher. If 2 noun phrases have the same negative class, then the one with the lower confidence is ranked higher. In the event where 2 or more noun phrases have the same classification and confidence value, ties are broken based on the rank of the passages from which the noun phrases originate. Ties are further broken based on the position of the noun phrase in the passage, i.e., noun phrases occurring earlier in the passage are ranked higher than those occurring later.

The top ranked noun phrase is then expanded with its neighboring words in the passage to make up a 50-byte (or 250-byte) answer string. The 2nd ranked noun phrase is then checked to ensure that it does not occur in the earlier chosen 50-byte (or 250-byte) answer strings. If it does, then the next ranked noun phrase is considered. If it does not, then the noun phrase is expanded to 50 bytes (or 250 bytes). This process of selecting and expanding noun phrases continues until finally 5 answer strings are chosen.

3 Evaluation

The training data we used to build our classifier consists of two parts. The first part is the set of 38 TREC-8 QA track development questions (Voorhees, 2000). Document collection for this set of questions is the same as in TREC-8 ad hoc task, namely the set of documents on TREC disk 4 and 5 minus the *Congressional Record* documents. In this development set, we have questions containing "what", "who", "how", "when", "where", and a few others that do not have any of these words. Answers to these questions are also provided by NIST together with the document id of the documents where the answers can be found. Answer phrases were then manual tagged in the specified documents and training examples were generated according to the scheme described in Section 2.

The second part of the training data is based on the questions and stories published by Remedia Publications (Hirschman et al., 1999). The document collection for this set consists of stories from grade 2 to 5 with a total of 115 stories. Each of the stories comes with five questions. However, not all the questions can be used for training, since some of the questions cannot be answered by any noun phrase in the associated story. We also ignore the "Why" questions, since their answers are typically not noun phrases. After removing the "Why" questions and questions without a noun-phrase answer, we have 398 questions left for training.

The collection of stories we used is the copy created by the MITRE group with each story manu-

Question type	Remedia	TREC-8 develop. set	TREC-8 test set
What	101	18	65
Who	98	7	48
How	0	6	31
When	95	3	18
Where	104	2	21
Others	0	2	15
Total	398	38	198

Table 1: Questions in the training and test set

ally annotated to indicate which sentence answers to each of the questions associated with the story. However, since in our training, we require the answer noun phrase instead of answer sentence, each story was then further hand-tagged to indicate the answer noun phrase. After this stage, the same scheme described in Section 2 was employed to generate training examples.

The test data we used is the set of questions in the official test set of TREC-8 QA Track (Voorhees, 2000). This set consists of 198 questions which has no overlap with the 38 development questions. Document collection is the same as TREC-8 ad hoc task. Table 1 shows the number of questions of each type in the training and test set.

Our evaluation is based on the official evaluation program and answer patterns released by the TREC-8 QA track organizer (Voorhees and Tice, 2000, SIGIR). In TREC-8, the official evaluation is done by human assessors. However, since we did not participate in the TREC-8 QA track, our runs have not been evaluated by the human assessors who judged the other TREC-8 official runs. Fortunately, as demonstrated in (Voorhees and Tice, 2000, SIGIR), evaluation based on the evaluation program and answer patterns gives comparable outcome as the evaluation done by human judges.

The evaluation metric we used is mean reciprocal rank (MRR), the same as that used in TREC-8. For each question, if the rank at which the first correct answer appears is k, then the question gets a score of $1/k$, where $k = 1, 2, 3, 4, 5$. If the correct answer is not found in the top 5 strings returned, then the question gets a score of 0. The overall score is the average MRR of all the 198 test questions.

The MRR score for our 50-byte run is 0.357, and 98 questions do not have any answer within the top 5 strings returned. For the 250-byte run, the MRR score is 0.525, and 71 questions do not have any answer within the top 5 strings returned. Table 2 shows the percentage of questions answered correctly (i.e., the answer appears within the top 5 strings) for the two runs, broken down according to the question types.

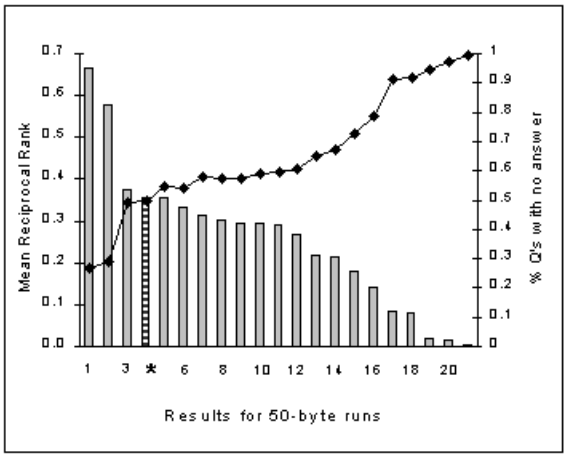

Figure 1: Results for 50-byte run

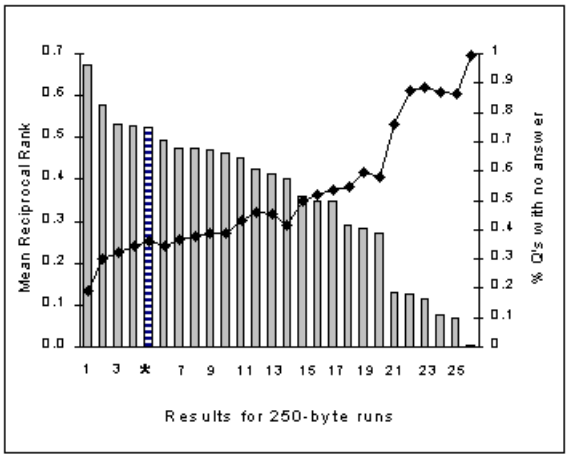

Figure 2: Results for 250-byte run

We compared our results with all the official runs submitted to TREC-8. All evaluation, for ours and for all other official runs, are made using the officially released evaluation program and answer patterns. Figure 1 shows our 50-byte run together with all TREC-8 50-byte runs. Figure 2 shows our 250-byte run together with all TREC-8 250-byte runs. In the 2 figures, vertical bars indicate MRR scores, whereas the dotted lines join the data points indicating the percentage of questions with no answers found in the top 5 returned strings. The shaded bars in the figures indicate the MRR scores of our system.

We have also performed statistical significance test between our runs and each of the TREC-8 official runs. The MRR score of each question is compared between two runs. The significance test results are summarized in Table 3. At 95% confidence level, our 50-byte run is significantly worse than 2 TREC-8 runs, better than 13, and no different from the rest. Our 250-byte run is significantly worse than

	What	Who	How	When	Where	Others
50-byte	38.5	62.5	48.4	66.7	52.4	46.7
250-byte	58.5	68.8	58.1	72.2	66.7	73.3

Table 2: Percentage of questions answered correctly

Runs	worse	no diff	better
50-byte	2	5	13
250-byte	1	6	18

Table 3: Significance test at 95% confidence level between our runs versus TREC-8 official runs

Features	MRR
minus QT	0.203
minus NPSC	0.256
minus QNP	0.329
minus DMWM	0.294
All	0.357

Table 4: Performance of our 50-byte run with one less feature versus all features

only one run, better than 18, and no different from the rest. This indicates that our QA system has achieved quite competitive accuracy.

It is interesting to note that our machine learning approach, which is based on a set of 4 simple features, can already give quite competitive performance. From Table 2, it can be seen that our system performs quite poorly on "What" and "How" questions. For "How" questions, the reason is probably the lack of sufficient training examples. There are only 6 "How" questions available for training from the TREC-8 development set, and none from the Remedia corpus. For "What" questions, the performance is low due to the fact that "What" questions can ask for just about anything, and our current set of semantic classes is not broad enough to capture the rich variety. In our future work, we will investigate improvements to overcome these deficiencies in our current system.

We also investigated the effect of training data size on the performance of our system, as well as the effectiveness of the features used. Figure 3 shows the learning curve of our 50-byte run when trained on 10%, 20%, ..., 100% of the available training data. The learning curve is obtained by averaging over 10 random trials. It appears that the performance of our system can be improved given a larger set of training data, which is encouraging.

Table 4 shows the performance of our 50-byte run if we remove one feature at a time. In all four cases, the performance of our system dropped when using only three features, as opposed to using all four features. This indicates that all four features contributed to the performance of our system.

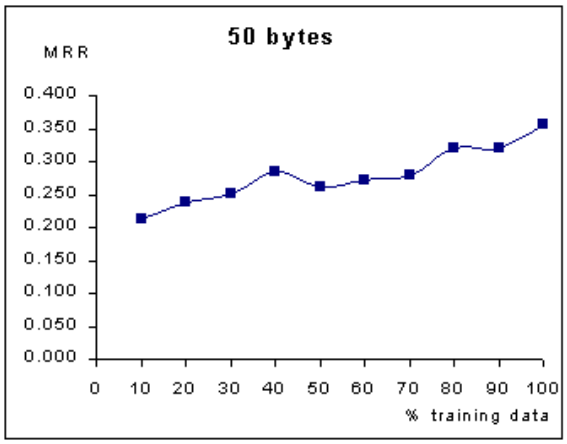

Figure 3: Learning curve for our 50-byte run

4 Related Work

Among the top performing systems at TREC-8 and TREC-9 QA track (Abney et al., 2000; Clarke et al., 2001; Cormack et al., 2000; Harabagiu et al., 2001; Hull, 2000; Moldovan et al., 2000; Singhal et al., 2000; Srihari and Li, 2000), most are not based on a machine learning approach. The exceptions are the work of (Ittycheriah et al., 2001; Prager et al., 2000; Prager et al., 2001). In (Ittycheriah et al., 2001), a maximum entropy approach is used to learn the type of a question from training questions. In our work, determining the question type is not based on learning, so this is something we would like to incorporate in the future. In the work of (Prager et al., 2000; Prager et al., 2001), logistic regression is used to learn the weights to combine scores for features, but their set of features is substantially different from ours.

In QA work on reading comprehension tests, again most are not based on a learning approach (Hirschman et al., 1999; Charniak et al., 2000; Riloff and Thelen, 2000). (Wang et al., 2000) attempted a machine learning approach, but with performance substantially lower than the other non-learning approaches.

Compared to our own previous work reported in (Ng et al., 2000), this paper differs in the following aspects. First, our previous work is only tested on reading comprehension of children stories, whereas the current paper scales up to answering questions based on real-world newspaper documents in TREC-8. Also, we now had to deal with question answer-

ing using a large text database, and not just answering questions posed to a single short story. In addition, the answers returned in our current work is a 50-byte or 250-byte string, and not a sentence. As such, the features used in this paper are centered around noun phrases, and not sentences. In addition, there is now no restriction on the type of questions asked. In particular, the current work can answer quantitative "How" questions not addressed in (Ng et al., 2000). Our current work also uses a different (simpler) set of 4 features, and does not rely on hand-tagged coreference information and named entity tags unlike (Ng et al., 2000).

5 Conclusion

In this paper, we presented a machine learning approach to question answering. The results indicate that such an approach is a promising way to build a state-of-the-art question answering system.

References

Steven Abney, Michael Collins, and Amit Singhal. 2000. Answer extraction. In *Proceedings of the 6th Applied Natural Language Processing Conference and the 1st Meeting of the North American Chapter of the Association for Computational Linguistics (ANLP-NAACL 2000)*, pages 296–301, Seattle, Washington.

Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. *Machine Learning*, 34(1-3):211–231.

Eugene Charniak, Yasemin Altun, Rodrigo de Salvo Braz, Benjamin Garrett, Margaret Kosmala, Tomer Moscovich, Lixin Pang, Changhee Pyo, Ye Sun, Wei Wy, Zhongfa Yang, Shawn Zeller, and Lisa Zorn. 2000. Reading comprehension programs in a statistical-language-processing class. In *Proceedings of the ANLP/NAACL 2000 Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, pages 1–5, Seattle, Washington.

Kenneth Church. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143.

C.L.A. Clarke, G.V. Cormack, D.I.E. Kisman, and T.R. Lynam. 2001. Question answering by passage selection (MultiText experiments for TREC-9). In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, Gaithersburg, Maryland.

G.V. Cormack, C.L.A. Clarke, C.R. Palmer, and D.I.E. Kisman. 2000. Fast automatic passage ranking (MultiText experiments for TREC-8). In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 735–741, Gaithersburg, Maryland.

Yoav Freund and Robert E. Schapire. 1996. Experiments with a new boosting algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 148–156.

Sanda Harabagiu, Dan Moldovan, Marius Pasca, Rada Mihalcea, Mihai Surdeanu, Razvan Bunescu, Roxana Girju, Vasile Rus, and Paul Morarescu. 2001. FALCON: Boosting knowledge for answer engines. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, Gaithersburg, Maryland.

Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. 1999. Deep Read: A reading comprehension system. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 325–332, College Park, Maryland.

David A. Hull. 2000. Xerox TREC-8 question answering track report. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 743–752, Gaithersburg, Maryland.

Abraham Ittycheriah, Martin Franz, Wei-Jing Zhu, and Adwait Ratnaparkhi. 2001. IBM's statistical question answering system. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, Gaithersburg, Maryland.

George A. Miller. 1990. WordNet: An online lexical database. *International Journal of Lexicography*, 3(4):235–312.

Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Richard Goodrum, Roxana Girju, and Vasile Rus. 2000. LASSO: A tool for surfing the answer net. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 175–183, Gaithersburg, Maryland.

Hwee Tou Ng, Leong Hwee Teo, and Jennifer Lai Pheng Kwan. 2000. A machine learning approach to answering questions for reading comprehension tests. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora. (EMNLP/VLC-2000)*, pages 124–132.

John Prager, Dragomir Radev, Eric Brown, Anni Coden, and Valerie Samn. 2000. The use of predictive annotation for questions answering in TREC8. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 399–409, Gaithersburg, Maryland.

John Prager, Eric Brown, Dragomir R. Radev, and Krzysztof Czuba. 2001. One search engine or two for questions-answering. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9)*, Gaithersburg, Maryland.

John Ross Quinlan. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, California.

Ellen Riloff and Michael Thelen. 2000. A rule-based

question answering system for reading comprehension tests. In *Proceedings of the ANLP/NAACL 2000 Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, pages 13–19, Seattle, Washington.

Amit Singhal, Steve Abney, Michiel Bacchiani, Michael Collins, Donald Hindle, and Fernando Pereira. 2000. AT&T at TREC-8. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 317–330, Gaithersburg, Maryland.

Rohini Srihari and Wei Li. 2000. Information extraction supported question answering. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 185–196, Gaithersburg, Maryland.

Ellen M. Voorhees. 2000. The TREC-8 question answering track report. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 77–82, Gaithersburg, Maryland.

Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 question answering track evaluation. In *Proceedings of the Eighth Text REtrieval Conference (TREC-8)*, pages 83–105, Gaithersburg, Maryland.

Ellen M. Voorhees and Dawn M. Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 200–207, Athens, Greece.

W. Wang, J. Auer, R. Parasuraman, I. Zubarev, D. Brandyberry, and M. P. Harper. 2000. A question answering system developed as a project in a natural language processing course. In *Proceedings of the ANLP/NAACL 2000 Workshop on Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, pages 28–35, Seattle, Washington.