A Machine Learning Approach to Answering Questions for Reading Comprehension Tests

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

Abstract

In this paper, we report results on answering questions for the reading comprehension task, using a machine learning approach. We evaluated our approach on the Remedia data set, a common data set used in several recent papers on the reading comprehension task. Our learning approach achieves accuracy competitive to previous approaches that rely on handcrafted, deterministic rules and algorithms. To the best of our knowledge, this is the first work that reports that the use of a machine learning approach achieves competitive results on answering questions for reading comprehension tests.

1 Introduction

The advent of the Internet has resulted in a massive information explosion. We need to have an effective and efficient means of locating just the desired information. The field of information retrieval (IR) is the traditional discipline that addresses this problem.

However, most of the prior work in IR deal more with document retrieval rather than "information" retrieval. This also applies to search engines on the Internet. Current search engines take a list of input words and return a ranked list of web pages that contain (or not contain) the given words. It is then left to the user to search through the returned list of web pages for the information that he needs. While finding the web pages that contain the desired information is an important first step, what an information seeker needs is often an answer to a question. That is, given a question, we want a system to return the exact answers to the question, and not just the documents to allow us to further search for the answers.

The need for question answering (QA) systems has prompted the initiation of the question answering track in TREC-8 (Voorhees and Tice, 2000) to address this problem. In the QA track, each participant is given a list of 200 questions, and the goal is to locate answers to these questions from a document database consisting of hundreds of thousands of documents (about two gigabytes of text). Each participant is to return a ranked list of the five best answer strings for each question, where each answer string is a string of 50 bytes (or 250 bytes) that contains an answer to the question. What, when, where, and who questions that have explicit answers given in some document in the database are emphasized, but not why questions.

In a related but independent effort, a group at MITRE has investigated question answering in the context of the reading comprehension task (Hirschman et al., 1999). The documents in this task are 115 children stories at grade two to five from Remedia Publications, and the task involves answering five questions (who, what, when, where, and why question) per story, as a measure of how well a system has understood the story. Each story has an average of 20 sentences, and the question answering task as formulated for a computer program is to select a sentence in the story that answers to a question. For about 10% of the questions, there is not a single sentence in the story that is judged to answer the question. Conversely, a question can have multiple correct answers, where each of several individual sentences is a correct answer. An example story from the Remedia corpus and its five accompanying questions are given in Figure 1. Each story has a title (such as "Storybook Person Found Alive!") and dateline (such as "ENGLAND, June, 1989") in the Remedia corpus.

* Leong Hwee Teo's current affiliation: Defence Medical Research Institute, Defence Science and Technology Agency, 1 Depot Road, Defence Technology Tower A, #19-05, Singapore 109679 tleonghw@dsta.gov.sg

Storybook Person Found Alive!

(ENGLAND, June, 1989) - Christopher Robin is alive and well. He lives in England. He is the same person that you read about in the book, Winnie the Pooh.

As a boy, Chris lived in a pretty home called Cotchfield Farm. When Chris was three years old, his father wrote a poem about him. The poem was printed in a magazine for others to read.

Mr. Robin then wrote a book. He made up a fairy tale land where Chris lived. His friends were animals. There was a bear called Winnie the Pooh. There was also an owl and a young pig, called a piglet. All the animals were stuffed toys that Chris owned. Mr. Robin made them come to life with his words. The places in the story were all near Cotchfield Farm.

Winnie the Pooh was written in 1925. Children still love to read about Christopher Robin and his animal friends. Most people don't know he is a real person who is grown now. He has written two books of his own. They tell what it is like to be famous.

1. Who is Christopher Robin?

2. What did Mr. Robin do when Chris was three years old?

3. When was Winnie the Pooh written?

4. Where did young Chris live?

5. Why did Chris write two books of his own?

Figure 1: A sample story and its five questions

Although both the TREC-8 QA task and the reading comprehension QA task are about question answering, there are a few differences in the two tasks. In TREC-8, the answer to a question can be from any of the hundreds of thousands of documents in the database, whereas for the reading comprehension task, the answer only comes from the short story associated with the question. Thus, while the TREC-8 QA task requires efficient indexing and retrieval techniques to narrow down to the documents that contain the answers, this step is largely not needed for the reading comprehension task. Also, an answer as defined in the TREC-8 QA task is a 50-byte or 250-byte answer string, whereas an answer is a complete sentence in the reading comprehension task. Another perhaps more subtle difference is that the questions formulated in both tasks have different motivation: for TREC-8, it is for the purpose of information-seeking, whereas for reading comprehension, it is for testing whether a system has "understood" the story. Hence, it may well be that the questions in TREC-8 can be expected to be more "cooperative", while those for reading comprehension can be uncooperative or even "tricky" in nature.

In this paper, we only address the question answering task in the context of reading comprehension, although we expect that the techniques we developed in this paper will be equally applicable to answering questions in an information-seeking context like that of TREC-8.

2 Related Work

The early work on question answering by (Lehnert, 1978) focused more on knowledge representation and inference issues, and the work was not targeted at answering questions from unrestricted text. Prior to 1999, the other notable research work on question answering that is designed to work on unrestricted text (from encyclopedia) is (Kupiec, 1993). However, no large scale evaluation was attempted, and the work was not based on a machine learning approach.

Fueled by the question answering track initiated in TREC-8 (Voorhees and Tice, 2000), there is a recent surge of research activities on the topic of question answering. Among the participants who returned the best scores at the TREC-8 QA track (Srihari and Li, 2000; Moldovan et al., 2000; Singhal et al., 2000), none of them uses a machine learning approach. One exception is the work of (Radev et al., 2000) at the TREC-8 QA track, which uses logistic regression to rank potential answers using a training set with seven features. However, their features are meant for the task of selecting more specific answer spans, and are different from the features we use in this paper. The TREC-8 QA test scores of (Radev et al., 2000) were also considerably lower than best QA test scores.

Because of the huge number of documents used in the TREC-8 QA track, the participants have to perform efficient document indexing and retrieval in order to tackle the complete QA task. It has been found that both the shallow processing techniques of IR, as well as the more linguistic-oriented natural language processing techniques are needed to perform well on the TREC-8 QA track. In contrast, for our current QA work on reading comprehension, because the answer for each question comes from the associated story, no sophisticated IR indexing and retrieval techniques are used.

Naturally, our current work on question answering for the reading comprehension task is most related to those of (Hirschman et al., 1999; Charniak et al., 2000; Riloff and Thelen, 2000; Wang et al., 2000). In fact, all of this body of work as well as ours are evaluated on the same set of test stories, and are developed (or trained) on the same development set of stories. The work of (Hirschman et al., 1999) initiated this series of work, and it reported an accuracy of 36.3% on answering the questions in the test stories. Subsequently, the work of (Riloff and Thelen, 2000) and (Charniak et al., 2000) improved the accuracy further to 39.7% and 41%, respectively. However, all of these three systems used handcrafted, deterministic rules and algorithms. In contrast, we adopt a machine learning approach in this paper.

The one notable exception is the work of (Wang et al., 2000), which attempted a machine learning approach to question answering for the same reading comprehension task. Unfortunately, out of the several machine learning algorithms they tried, the best approach (using neural network learning) only managed to obtain an accuracy of 14%. This work casts doubt on whether a machine learning approach to question answering can achieve accuracy competitive to the handcrafted rule-based approach. Our current work attempts to address exactly this issue.

3 A Machine Learning Approach

In this section, we present our machine learning approach to question answering. We have successfully implemented a question answering system based on this approach. Our system is named AQUAREAS (Automated QUestion Answering upon REAding Stories). The advantage of a machine learning approach is that it is more adaptable, robust, flexible, and maintainable. There is no need for a human to manually engineer a set of handcrafted rules and continuously improve or maintain the set of rules.

For every question, our QA task requires the computer program to pick a sentence in the associated story as the answer to that question. In our approach, we represent each question-sentence pair as a feature vector. Our goal is to design a feature vector representation such that it provides useful information to a learning algorithm to automatically build five classifiers, one for each question type. In prior work (Hirschman et al., 1999; Charniak et al., 2000; Riloff and Thelen, 2000) the number and type of information sources used for computation is specific to and different for each question type. In AQUAREAS, we use the same set of features for all five question types, leaving it to the learning algorithm to identify which are the useful features to test for in each question type.

The machine learning approach comprises two steps. First, we design a set of features to capture the information that helps to distin-

guish answer sentences from non-answer sentences. Next, we use a learning algorithm to generate a classifier for each question type from the training examples.

Each training example or test example is a feature vector representing one question-sentence pair. Given a question q in a story, one positive example is generated from each sentence s that is marked (by the MITRE group) as an answer to q, and the question q itself. For negative training examples, all other sentences that are not marked as answers to a question q can be used. In AQUAREAS, we use all other sentences that are marked as answers to the questions other than q in the same story to generate the negative examples for question q. This also helps to keep the ratio of negative examples to positive examples from becoming too high.

3.1 Feature Representation

Our feature representation was designed to capture the information sources that prior work (Hirschman et al., 1999; Charniak et al., 2000; Riloff and Thelen, 2000) used in their computations or rules. We hypothesize that given equivalent information sources, a machine learning approach can do as well as a system built using handcrafted rules. Our feature vector consists of 20 features.

- **Diff-from-Max-Word-Match (DMWM)**

  The possible values for this feature are 0, 1, 2, 3, ... For a given question q and a sentence s, the value for this feature is computed by first counting the number of matching words present in q and s, where two words match if they have the same morphological root. This gives the raw word match score m for the question-sentence pair q and s. Next, we find the highest raw word match score M over all sentences s_i in the story and q. The value of this feature DMWM for the question-sentence pair q and s is $M - m$.[1]

  Intuitively, a feature value of 0 is the best, indicating that for that question-sentence pair q and s, they have the most number of matching words in the story, when comparing q with all sentences s_i in the same story.

  In the case where there are zero matching words between a question q and all sentences s_i in a story (i.e., $M = 0$), then this DMWM feature will be assigned 0 for all question-sentence pairs q and s_i in the story. To avoid such a situation where a value of 0 is also assigned for this feature even when there are no matching words, we instead assign a very large value (200) to this feature for such cases.

- **Diff-from-Max-Verb-Match (DMVM)**

  The possible values for this feature are 0, 1, 2, 3, ... The value for this feature is computed in exactly the same way as DMWM, except that we only count main verb matches between a question and a sentence, excluding verbs whose morphological roots are "be", "do" or "have". (Such verbs tend not to carry as much "semantic" information.)

- **DMWM-Prev, DMVM-Prev, DMWM-Next, DMVM-Next**

  The possible values for each of these features are 0, 1, 2, 3, ... Their computation is similar to DMWM and DMVM, except that they are computed from the question q and the sentence s_{-1} (the sentence preceding the current sentence s in consideration), in the case of DMWM-Prev and DMVM-Prev, and the question q and the sentence s_{+1} (the sentence following s) in the case of DMWM-Next and DMVM-Next. For the title and dateline of a story, we take them as having no previous or next sentences. For the first sentence in the body of a story, there is no previous sentence and likewise for the last sentence in the body of a story, there is no next sentence. For all such cases, we give a raw word/verb match score m of 0 in the computation.

  We designed these 4 features to capture information that will be helpful to the why questions, since it has been observed in prior work (Charniak et al., 2000; Riloff and Thelen, 2000) that the answer

[1] In an earlier version of AQUAREAS, we simply used the raw word match score m as the feature. However, the learned classifiers did not perform well. We suspect that the absolute raw word match score m may not matter as much as whether a sentence has the highest raw word match score M in a story (relative to other sentences in the same story). We address this deficiency in our reformulated difference-from-maximum computation.

sentence to a why question tends to follow (or precede) the sentence in the story that has the most number of word matches with the question.

- **Sentence-contains-Person, Sentence-contains-Organization, Sentence-contains-Location, Sentence-contains-Date, Sentence-contains-Time**

  The possible values for each of these features are true or false. To compute these feature values for a sentence, we used the Remedia corpus provided by MITRE which has been hand-tagged with named entities. If a sentence contains at least one word tagged with the named entity person, then the feature Sentence-contains-Person will be assigned the value true. Its value is false otherwise. Similarly for the other four named entities organization, location, date, and time.

- **Coreference information**

  Coreference information does not contribute any new features, but rather it may change the values assigned to the five features Sentence-contains-Person, Sentence-contains-Organization, ... By using the Remedia corpus provided by MITRE which has been hand-tagged with coreference chains of noun phrases, we can propagate a named entity tag across all noun phrases in the same coreference chain. We then utilize the revised propagated named entities to assign values to the five named entity features for a sentence. The effect of using coreference information is that for some sentence, a named entity feature may have its value changed from false to true. This occurs when, for instance, a pronoun "he" in a sentence corefers to a noun phrase "Mr. Robin" and inherits the named entity tag person from "Mr. Robin" in the same coreference chain.

- **Sentence-is-Title, Sentence-is-Dateline**

  The possible values for each of these features are true or false. If a sentence is the title of a story, then the feature Sentence-is-Title will be assigned the value true. Its value is false otherwise. Similarly, the feature Sentence-is-Dateline applies to a sentence which is the dateline of a story. It has been observed in prior work (Charniak et al., 2000; Riloff and Thelen, 2000) that such sentences may be more likely to be the answer sentences to some question type (for example, dateline can answer to when questions).

- **keywords in sentences**

  The idea behind the use of this group of features is that certain words in a sentence may provide strong clues to the sentence being the answer to some question type. For instance, the preposition "in" (such as "...in the United States, ...") may be a strong clue that the sentence is an answer to a where question.

  We devised an automated procedure to find such words. For each of the five question types, we collect all the sentences in the training stories that answer to the question type. Any word (in its morphological root form) that occurs at least 3 times in this set of sentences is a possible candidate word. For each candidate word, we compute the following correlation metric C (Ng et al., 1997):

$$C = \frac{(N_{r+}N_{n-} - N_{r-}N_{n+})\sqrt{N}}{\sqrt{(N_{r+}+N_{r-})(N_{n+}+N_{n-})(N_{r+}+N_{n+})(N_{r-}+N_{n-})}}$$

  where N_{r+} (N_{n+}) is the number of training story sentences that answer (do not answer) to the question type and in which the word w occurs, and N_{r-} (N_{n-}) is the number of training story sentences that answer (do not answer) to the question type and in which the word w does not occur. $N = N_{r+} + N_{r-} + N_{n+} + N_{n-}$.

  Note that the correlation metric C is the square root of the χ^2 metric. A candidate word that has high positive C value is a good clue word. If such a word occurs in a sentence, then the sentence is likely to answer to the question type.

  For each question type, we find one word that has the highest positive C value for that question type. The following five words ("name", "call", "year", "in", and "to") are found automatically in this way for the five question types who, what, when, where, and why, respectively. One feature is then formed for each word: whether a sentence contains the word "name", whether a sentence contains the

word "call", etc. The possible values for these features are true or false.

Note that this list of keywords is determined automatically from the training stories only, *without* looking at the test stories.

- **keywords in questions**

  It has been observed in the work of (Riloff and Thelen, 2000) that certain words in a when or where question tend to indicate that the dateline is an answer sentence to the question. The words used in (Riloff and Thelen, 2000) are "happen", "take place" "this", "story".

  In our work, we attempted to discover these words automatically, using the correlation metric. The method is the same as what we used to discover the keywords in sentences, except that N_{r+} (N_{n+}) is the number of training story questions that have (do not have) dateline as an answer to the question, and in which the word w occurs, and N_{r-} (N_{n-}) is the number of training story questions that have (do not have) dateline as an answer to the question and in which the word w does not occur.

  We again picked the word with the highest positive C value for each question type. Only two words ("story and "this") are found, for the when and where question, respectively. For the other question types, either the dateline was never an answer sentence to the question type, or that no candidate words occur at least three times in the training story questions.

  We then form one feature for each word: whether a question contains the word "story", and whether a question contains the word "this". The possible values for these features are true or false. Again, these two keywords are determined from the training stories only, *without* looking at the test stories. It is interesting to note that the words automatically determined by our procedure are also part of those words found manually in the prior work of (Riloff and Thelen, 2000).

3.2 Building Classifiers

The next step is to use a machine learning algorithm to learn five classifiers from the training examples, one classifier per question type. The learning algorithm we used in AQUAREAS is C5, a more recent version of C4.5 (Quinlan, 1993).

For each test example, the classifier will decide if it is positive (an answer) or negative (not an answer) with a confidence value. We pick as the answer to the question the sentence whose feature vector was classified positive with the highest confidence, or in the absence of such, the sentence classified negative with the lowest confidence. AQUAREAS breaks ties in favor of the sentence appearing earlier in the story.

C5 accepts parameters that affect its learning algorithm. The following three parameters were used in AQUAREAS to achieve better performance. m avoids over-fitting the training data by specifying that a minimum number of m examples must follow a decision tree branch. t specifies the maximum number of decision trees used in adaptive boosting to determine the final decision through voting. n/p cost influences C5 to avoid false negatives (or false positives).

4 Evaluation

To evaluate our learning approach, we trained AQUAREAS on the same development set of stories and tested it on the same test set of stories as those used in all past work on the reading comprehension task (Hirschman et al., 1999; Charniak et al., 2000; Riloff and Thelen, 2000; Wang et al., 2000). Specifically, the set of stories used are published by Remedia Publications. We used the same softcopy version created by the MITRE group, and the material includes manual annotations of named entities and coreference chains as done by the MITRE group.

The training set consists of 28 stories from grade 2 and 27 stories from grade 5. The test set consists of 30 stories from grade 3 and 30 stories from grade 4. Within the 60 test stories, there are 59 who questions, 61 what questions, 60 when questions, 60 where questions, and 60 why questions, for a total of 300 test questions.

The scoring metric that we used for evaluation is HumSent, which is the percentage of test questions for which AQUAREAS has chosen a correct sentence as the answer. This metric is originally proposed by (Hirschman et al., 1999). The correct answer sentences are chosen manually by the MITRE group. Although there were a few other scoring met-

rics originally proposed in (Hirschman et al., 1999), all the metrics were found to correlate well with one another. As such, all subsequent work (Charniak et al., 2000; Riloff and Thelen, 2000; Wang et al., 2000) uses HumSent as the main scoring metric, and it is also the scoring metric that we adopted in this paper.

Based on the complete set of 20 features described in the previous section, we trained one classifier per question type. For each question type, we uniformly use the same, identical set of features. The following learning parameters were found to give the best HumSent accuracy and were uniformly used in generating all the decision tree classifiers for all question types reported in this paper: $m = 37$, $t = 7$, and n/p cost $= 1.2$. Using a large m results in simpler decision trees. $t = 7$ results in the use of boosting with multiple decision trees. Since there are a lot more negative training examples compared to positive training examples (ratio of approximately 4:1), there is a tendency to generate a default tree classifying all training examples as negative (since the accuracy of such a tree is already quite good – about 80% on our skewed distribution of training examples). Setting n/p cost at 1.2 will make it more costly to misclassify a positive training example as negative, and thus more costly to generate the default tree, resulting in better accuracy.

We achieved an overall HumSent accuracy of 39.3% on the 300 test questions. The breakdown into the number of questions answered correctly per question type is shown in the first row of Table 1. Our results indicate that our machine learning approach can achieve accuracy comparable with other approaches that rely on handcrafted, deterministic rules and algorithms. For comparison, the HumSent scores reported in the work of (Hirschman et al., 1999), (Charniak et al., 2000), (Riloff and Thelen, 2000), and (Wang et al., 2000) are 36.3%, 41%, 39.7%, and 14%, respectively.

Figure 2 shows the sequence of decision trees generated via boosting for the when question type. All the trees look reasonable and intuitive. The first tree states that if the Diff-from-Max-Word-Match is zero (i.e., the sentence has the highest number of word match to the question), then the sentence is an answer. Otherwise, the classifier tests for whether the sentence contains a date. If it does, then the sentence is an answer, else it is not an answer. The second tree is a default tree that just classifies any sentence as not an answer. The rest of the trees similarly test on features that we intuitively feel are indicative of whether a sentence answers to a when question.

To investigate the relative importance of each type of features, we remove one type of features at a time and observe its impact on HumSent accuracy. The resulting drop in accuracy is tabulated in the remaining rows of Table 1. The rows are ordered in decreasing overall HumSent accuracy.

As expected, removing the word match feature causes the largest drop in overall accuracy, and the accuracy decline affects all question types. Removing the five named entity features also causes a large decline, affecting mainly the who, when, and where questions. Named entities are useful for answering these question types, since who typically asks for a person (or organization), when asks for date or time, and where asks for location. What is perhaps a little surprising is that the seven automatically discovered keywords are also found to be very important, and removing these seven features causes the second largest decline in overall HumSent accuracy.

Coreference is found to affect the who, when, and where questions, as expected. The previous and next word/verb matches cause the largest decline for why questions, dropping the number of correctly answered why questions to 3. Removing verb match also causes a 3% drop in overall accuracy, while dateline and title only affect the when questions.

In our future work, we plan to investigate other potential knowledge sources that may further improve accuracy. We also plan to investigate the use of other supervised machine learning algorithms for this problem.

5 Conclusion

In summary, we reported in this paper the results on answering questions for the reading comprehension task, using a machine learning approach. We evaluated our approach on the Remedia data set, a common data set used in several recent papers on the reading comprehension task. Our learning approach achieves accuracy competitive to previous approaches that rely on handcrafted, deterministic rules and algorithms. To the best of our knowledge, this is the first work that reports that the use of a machine learning approach achieves

Features	Who	What	When	Where	Why	All
All	31	21	27	31	8	118 (39.3%)
− title & dateline	31	21	19	31	8	110 (36.7%)
− DMVM	24	21	31	25	8	109 (36.3%)
− prev & next	27	21	26	25	3	102 (34.0%)
− coreference	27	21	24	22	7	101 (33.7%)
− named entities	24	21	21	23	6	95 (31.7%)
− keywords	27	21	19	17	9	93 (31.0%)
− DMWM	29	6	20	24	5	84 (28.0%)

Table 1: Accuracy using different set of features

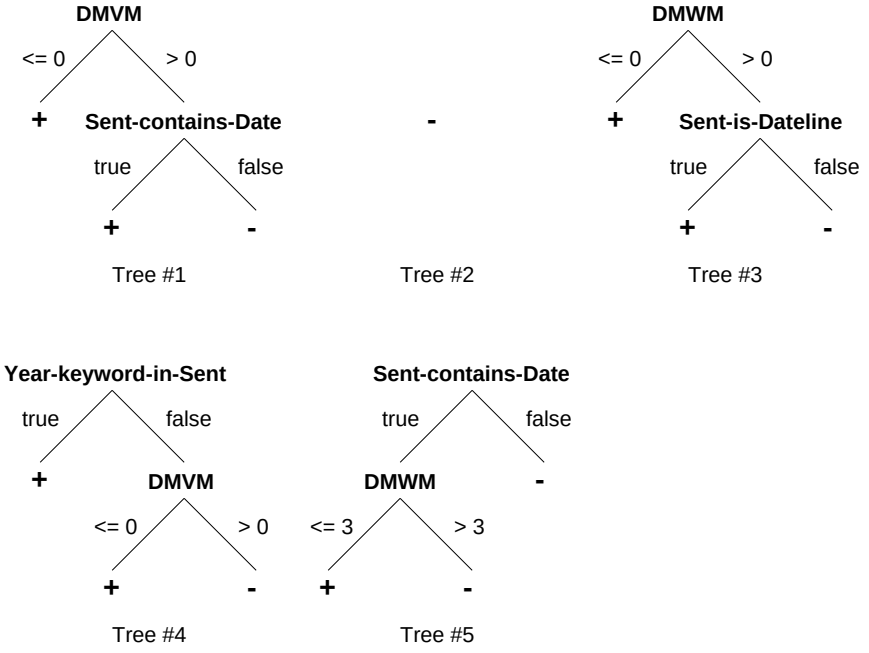

Figure 2: The classifier learned for the when question type

competitive results on answering questions for reading comprehension tests.

6 Acknowledgements

Thanks to Marc Light and Eric Breck of MITRE Corporation for assistance in providing the annotated Remedia corpus.

References

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.

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.

Julian Kupiec. 1993. MURAX: a robust linguistic approach for question answering using an online encyclopedia. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 181–190, Pittsburgh, Pennsylvania.

Wendy Lehnert. 1978. *The Process of Question Answering*. Lawrence Erlbaum Associates, Hillsdale, NJ.

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), Gaithersburg, Maryland.

Hwee Tou Ng, Wei Boon Goh, and Kok Leong Low. 1997. Feature selection, perceptron learning, and a usability case study for text categorization. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 67–73, Philadelphia, Pennsylvania.

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

Dragomir R. Radev, John Prager, and Valerie Samn. 2000. Ranking suspected answers to natural language questions using predictive annotation. In *Proceedings of the Sixth Applied Natural Language Processing Conference*, pages 150–157, Seattle, Washington.

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)*, Gaithersburg, Maryland.

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

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.