Information Extraction with Term Frequencies*

T. R. Lynam C. L. A. Clarke G. V. Cormack

Computer Science
University of Waterloo
Ontario, Canada
mt@plg.uwaterloo.ca

1. INTRODUCTION

Every day, millions of people use the internet to answer questions. Unfortunately, at present, there is no simple and successful means to consistently accomplish this goal. One common approach is to enter a few terms from a question into a Web search system and scan the resulting pages for the answer, a laborious process. To address this need, a question answering (QA) system was created to find and extract answers from a corpus. This system contains three parts: a parser for generating question queries and categories, a passage retrieval element, and an information extraction (IE) component. The extraction method was designed to elicit answers from passages collected by the information retrieval engine. The subject of this paper is the information extraction component. It is based on the premise that information related to the answer will be found many times in a large corpus like the Web.

The system was applied to the Question Answering Track at TREC-9 and achieved the second best results overall[3]. The information extraction and parsing components were new for TREC-9; the TREC-8 system solely used passage retrieval[4]. Each new component yielded greater than 10% improvement in mean reciprocal rank, TREC's standard evaluation measure.

In the sections that follow, the extraction component is described and evaluated according to its contribution to the system's effectiveness. In particular, this paper investigates the contribution of a voting scheme favouring terms found in many candidate passages.

2. BACKGROUND

Architecturally, the question answering system is simple. First the parser analyses the question and generates a query for the passage retrieval component. It also provides selection rules for the information extraction component. Next, the passage retrieval component executes the query over the target corpus and retrieves a ranked list of passages for the answer IE component to process. Thirdly, the information extraction component finds the answers' extracts in the passages retrieved.

The parser is a probabilistic version of Earley's algorithm. It determines all possible parses of the grammar and selects the most probable. The grammar contains only 80 production rules[3].

.

The passage retrieval component collects arbitrary substrings of a document in the corpus. These substrings are considered passages and given a score. Passage scores are based on the terms contained in the query and the passage length. Passages with a length of one thousand words were retrieved in the TREC-9 system.

The information extraction component locates possible answers in the top ten passages. It then selects the best answer extracts of a predetermined length.

The overall approach of question analysis followed by IR succeeded by IE is nearly universal in QA systems[1, 2, 5, 6, 7, 8, 9]. The TREC-9 question answering track required the QA system to find solutions to 693 questions. Two different runs were judged: 50- and 250-byte answer extracts. Question answering systems were evaluated by the mean reciprocal answer rank (MRR). Five passages of the desired length are evaluated in order. The score is based on the rank of the first correct passage according to the formula:

$$MRR = \frac{1}{\#questions} \sum_{i=1}^{\#questions} \frac{1}{answer_i\ rank}$$

If the answer is found at multiple ranks, the best (lowest) rank will be used. If an answer is not found in the top five, the score for that particular question is zero.

The TREC-9 results reveal the improvements of the new components added to the system. The TREC-8 system was used as a baseline. With the combination of the parse-generated queries and the information extraction components, there is a total improvement of 106% and 25% for 50- and 250-byte runs respectively. The information extraction element has a greater impact when the answer is shorter as seen in Table 1.

3. TERM FREQUENCY ALGORITHM

The algorithm requires a set of passages that are likely to contain an answer, and a category for each question. This algorithm is similar to the information extraction technique used in the GuruQA system[8]. The key to the algorithm is using term frequencies to give individual terms a score. Important information is uncovered by looking at repeated terms in a set of passages. In addition, terms are scored based on their recurrence in the corpus. The system applies very simple patterns to discover individual words or numbers, allowing the evaluation of the term's frequency. This method proceeds in the following sequence:

1. Simplify the question category from the parser output.
2. Scan the passages for patterns matching the question category.

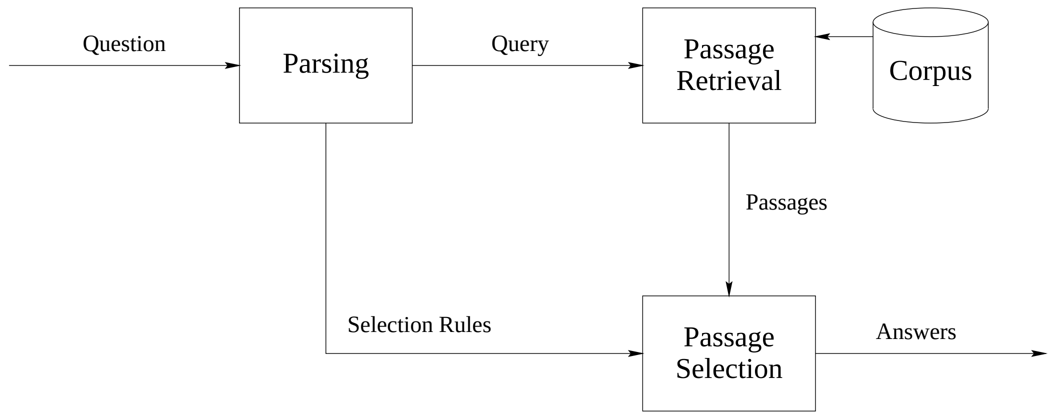

Figure 1: Overview of QA processing.

Table 1: Mean reciprocal ranks using TREC-9 evaluation

	50-byte answer MRR		250-byte answer MRR	
baseline	0.189		0.407	
parse-generated queries improvement	0.191	(+1%)	0.464	(+14%)
information extraction improvement	0.357	(+89%)	0.467	(+15%)
TREC-9 system	0.390	(+106%)	0.507	(+25%)

3. Assign each possible answer term an initial weight based on its rareness.

4. Modify each term weight depending on its distance from the centre and rank of the passage.

5. Select the (50-byte or 250-byte TREC 9 format) answer that maximizes the sum of the terms' weight found within the passage.

6. Set all terms' weight in the selected answer to zero.

7. Repeat steps 5 and 6 until five answers are selected.

The initial procedure simplifies the answer categories. The algorithm utilizes the question classification given by the parser in the following categories: *Proper* (person, name, company), *Place* (city, country, state), *Time* (date, time of day, weekday, month, duration, age), *How* (much, many, far, tall, etc.). The latter category is divided into sub-categories for monetary values, numbers, distances and other methods of measurement.

Next, the passages are scanned using the patterns for the given question classification. The purpose of the patterns is to narrow the number of possible answers which will increases the performance. It is important to note that the patterns do not contribute to the terms' weight. These simple patterns are regular expressions that have been hand-coded. For example, the pattern for *Proper* is [ˆA-Za-z][A-Z][A-Za-z][ˆA-Za-z0-9], which matches a capital letter followed by one or more letters surrounded by white space or punctuation. Each word in the passage either matches a pattern or not. Patterns do not stretch over more than one word. In the passage ”Bank of America” only ”Bank” and ”America” would be considered possible answers. The algorithm can find the correct answer ”Bank of America” by determining that ”Bank” and ”America” should be in the answer. When question classification is unknown, the term frequency for all words in the passages is computed. The system was evaluated using no question classification and still achieved a MRR of 0.338. With no classification, only the term frequency equation is utilized to evaluate answers. This confirms the power of the term frequency equation (1). The patterns for each question classification are very naive so in theory, if the patterns were improved the entire system would also improve.

Thirdly, the terms are differentiated by assigning each term a weight. The term weight is related to the term's rareness. The rarer the term, the higher the term's value. The power of the information extraction component is almost entirely derived from this step. Each term's weight is calculated by the following formula:

$$w_t = c_t log(N/f_t) \qquad (1)$$

where f_t is the number of times the term is in the corpus, c_t is the number of times the term is in the set of passages, and N is the total number of terms in the corpus. Knowing the term's corpus frequency is important; however, the strength of the formula is drawn from the multiple occurrences of terms appearing in the retrieved passages. An answer extract containing ”Bank of America” will most likely be selected if ”Bank” and ”America” have high term frequency values. Essentially, this calculation employs the corpus term frequency in conjunction with a voting scheme. The equation will reveal the rarest term in the corpus that occurs most often in the passages retrieved.

The fourth step modifies the term weight depending on its location. The centre of the passages is the centre of the query terms' locations. As a possible answer's distance from the centre increases its relation to the query, terms decrease. To utilize this information, the term weight is modified in conjunction with its distance from the centre of the passage. The farther from the centre, the more the term weight is decreased. The term value is then modified according to the passage ranking in which it was found; the lower the ranking, the more the term weight is decreased. Step four is important because it distinguishes duplicate terms depending on each term's position. This means that if there are many duplications of a possible answer each one will have a different term weight. For

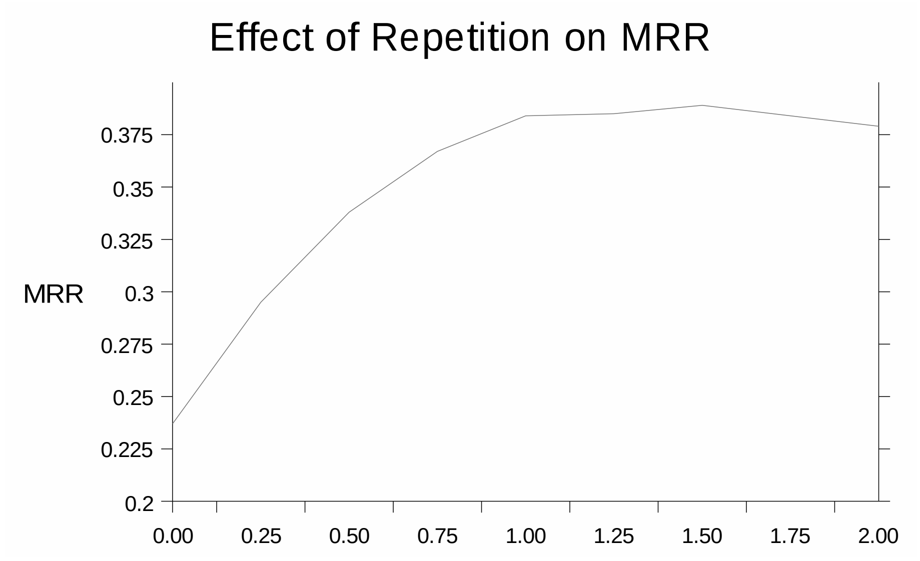

Figure 2: Significance of repetition in term frequency equation.

example, the term ”Bank” found in the best passage would have a higher term weight than a ”Bank” term found in a lower ranking passage.

For TREC-9, the system was required to produce 50- and 250-byte substrings. Each substring is assigned a score equal to the sum of the terms' weight within it. The best answer is the substring of the required length with the highest score. The weight of all the terms appearing in the answer substring is reduced to zero (step six). The final step is the selection of the next best substring; this process repeats until the number of desired substrings is fulfilled. Reducing the terms' weight to zero allows for distinction between each of the answers, eliminating answers that are almost the same. When a term is part of a phrase like “knowing is half the battle” the terms in the phrase will usually appear together in the retrieved passages. This means the phrase would be selected if “knowing” , “half”, and “battle” scored highly.

The idea behind the algorithm is to evaluate potential answers in the passages retrieved using the term frequency equation. The question classification patterns are used to limit the number of possible answers evaluated, which heightens accuracy. The algorithm will select phrases even if all the words are not possible answers. The term frequency algorithm does not need to know the answer classification to perform proficiently. This is a very robust method to extract answers, though knowing the question classification does improve the system's mean reciprocal rank considerably.

In the future, term frequencies may be used in combination with Natural Language Processing (NLP) techniques such as a name entity tagging to further enhance the system's results.

4. RESULTS

In a large corpus there is duplicate or supporting information for almost any given question. The term frequency formula utilizes this knowledge, through two simple premises: the more a term is repeated, the more conceivable it is the correct answer, and the less likely a term appears by chance, the more probable it is also correct.

The duplication component's importance in formula (1) can be evaluated by modifying the value of α in the term frequency equation:

$$w_t = c_t^{\alpha} log(N/f_t) \tag{2}$$

Figure 2 demonstrates the value that duplicate information in the passages has on the result by modifying α.

The graph reveals that as the importance of duplicate terms increases, the performance of the system strengthens. By eliminating the repetition part of the equation ($\alpha = 0$) the system only achieves a mean reciprocal rank of 0.237. As expected and demonstrated in the graph, the value of this part of the formula reaches a maximum before decreasing the overall system's accuracy.

5. CONCLUSION

Overall, the information extraction component improves the question answering system. Notably, the term frequency algorithm does not require information regarding the structure or grammar of a natural language; therefore the algorithm may be use in many natural languages. The term frequency algorithm can even extract answers when the question's meaning is completely unknown. Having an elementary and reliable way to evaluate each term in a set of passages is useful. One possibility is to add highly weighted terms to the original query.

In theory, as the corpus size expands, the performance of the system should increase as more duplicate information will become available. Finally, the initial value of the term frequency algorithm is beneficial to the overall system and future applications of question answering.

6. REFERENCES

[1] E. Breck, J. Burger, D. House, M. Light, and I. Mani. Question answering from large document collections. In *1999 AAAI Fall Symposium on Question Answering Systems*, North Falmouth, MA, 1999.

[2] C. Cardie, V. Ng, D. Pierce, and C. Buckley. Examining the role of statistical and linguistic knowledge sources in a general-knowledge question-answering system. In *Sixth Applied Natural Language Processing Conference*, pages 180–187, 2000.

[3] C. L. A. Clarke, G. V. Cormack, D. I. E. Kisman, and T. R. Lynam. Question answering by passage selection. In *9th Text REtrieval Conference*, Gaithersburg, MD, 2000.

[4] G. V. Cormack, C. L. A. Clarke, C. R. Palmer, and D. I. E. Kisman. Fast automatic passage ranking. In *8th Text REtrieval Conference*, Gaithersburg, MD, November 1999.

[5] S. M. Harabagiu and S. J. Maiorano. Finding answers in large collections of texts: Paragraph indexing + abductive inference. In *1999 AAAI Fall Symposium on Question Answering Systems*, pages 63–71, North Falmouth, MA, 1999.

[6] E. Hovy, U. Hermjakob, C.-Y. Lin, M. Junk, and L. Gerber. The Webclopedia. In *9th Text REtrieval Conference*, Gaithersburg, MD, 2000.

[7] A. Ittycheriah, M. Franz, W.-J. Zhu, and A. Ratnaparkhi. IBM's statistical question answering system. In *9th Text REtrieval Conference*, Gaithersburg, MD, 2000.

[8] D. R. Radev, J. Prager, and V. Samn. Ranking suspected answers to natural language questions using predictive annotation. In *6th Conference on Applied Natural Language Processing*, Seattle, May 2000.

[9] W. A. Woods, S. Green, P. Martin, and A. Houston. Halfway to question answering. In *9th Text REtrieval Conference*, Gaithersburg, MD, 2000.