論文 / 著書情報
Article / Book Information

Title(English)	Question Answering as a Classification Task
Authors(English)	Edward Whittaker, Sadaoki Furui
Citation(English)	Symposium on Large-Scale Knowledge Resourses(LKR2005),, Vol. , No. , pp. 39-42
発行日 / Pub. date	2005, 3

Proc. Proc. Symposium on Large-Scale Knowledge Resources (LKR 2005), Tokyo, pp. 39-42 (2005)

QUESTION ANSWERING AS A CLASSIFICATION TASK

Edward Whittaker and Sadaoki Furui

Dept. of Computer Science
Tokyo Institute of Technology
2-12-1, Ookayama, Meguro-ku
Tokyo 152-8552 Japan
{edw,furui}@furui.cs.titech.ac.jp

ABSTRACT

In this paper we treat question answering (QA) as a classification problem. Our motivation is to build systems for many languages without the need for highly tuned linguistic modules. Consequently, word tokens and web data are used extensively but no explicit linguistic knowledge is incorporated. A mathematical model for answer retrieval, answer classification and answer length prediction is derived. The TREC 2002 QA task is used for system development where a confidence weighted score (CWS) of 0.551 is obtained. Performance is evaluated on the factoid questions of the TREC 2003 QA task where a CWS of 0.419 is obtained which is in the mid-range of contemporary QA systems on the same task.

1. INTRODUCTION

Question answering (QA), particularly in the style of the Text Retrieval Conference (TREC) evaluations, has attracted significantly increased interest over recent years during which time "standard architectures" have evolved. More recently, there have been attempts to diverge from the highly-tuned, linguistic approaches towards more data-driven, statistical approaches [1, 2, 3, 4, 5, 6]. In this paper, we present a new, general framework for question answering and evaluate its performance on the TREC 2003 QA task [7].

QA, especially in the context of the Web, has been cited recently as the next-big-thing in search technology [8]. Exploitation of the huge domain coverage and redundancy inherent in web data has also become a theme in many TREC participants' systems e.g. [2, 6]. Redundancy in web data may be seen as effecting data expansion as opposed to the query expansion techniques and complex linguistic analysis often necessary in answering questions using a fixed corpus, such as the AQUAINT corpus [9] where there are typically relatively few answer occurrences for any given question.

The availability of large amounts of data, both for system training and answer extraction logically leads to examining statistical approaches to question answering. In [1] a number of statistical methods is investigated for what was termed bridging the lexical gap between questions and answers. In [4] a maximum-entropy based classifier using several different features was used to determine answer correctness and in [5] performance was compared against classifying the actual answer. A statistical noisy-channel model was used in [3] in which the distance computation between query and candidate answer sentences is performed in the space of parse trees. In [6] the lexical gap is bridged using a statistical translation model. Of these, our approach is probably most similar to [6] and the re-ranker in [5]. Statistical approaches still underperform the best TREC systems but have a number of potential advantages over highly tuned linguistic methods including robustness to noisy data, and rapid development for new languages and domains.

In this paper we take a statistical, noisy-channel approach and treat QA as a whole as a classification problem. We present a new mathematical model for including all manner of dependencies in a consistent manner that is fully trainable if appropriate training data is available. In doing so we largely remove the need for ad-hoc weights and parameters that are a feature of many TREC systems. Our motivation is the rapid development of data-driven QA systems in new languages where the need for highly tuned linguistic modules is removed. Apart from our new model for QA there are two major differences between our approach and many contemporary approaches to QA. Firstly, we only use word tokens in our system and do not use WordNet, named-entity (NE) extraction, or any other linguistic information e.g from semantic analysis or from question parsing. Secondly, we use a search engine to find relevant web documents and extract answers from the documents as a whole, rather than retrieving smaller text units such as sentences prior to determining the answers.

2. CLASSIFICATION TASK

It is clear that the answer to a question depends primarily on the question itself but also on many other factors such as the person asking the question, the location of the person, what questions the person has asked before, and so on. Although such factors are clearly relevant in a real-world scenario they are difficult to model and also to test in an off-line mode, for example, in the context of the TREC evaluations. We therefore choose to consider only the dependence of an answer A on the question Q, where each is considered to be a string of l_A words $A = a_1, \ldots, a_{l_A}$ and l_Q words $Q = q_1, \ldots, q_{l_Q}$, respectively. In particular, we hypothesize that the answer A and its length l_A depend on two sets of features $W = \mathcal{W}(Q)$ and $X = \mathcal{X}(Q)$ as follows:

$$P(A \mid Q) = P(A, l_A \mid W, X), \tag{1}$$

where $W = w_1, \ldots, w_{l_W}$ can be thought of as a set of l_W features describing the "question-type" part of Q such as *when*, *why*, *how*, etc. and $X = x_1, \ldots, x_{l_X}$ is a set of l_X features comprising the "information-bearing" part of Q i.e. what the question is actually

about and what it refers to. For example, in the questions, *Where was Tom Cruise married?* and *When was Tom Cruise married?* the information-bearing component is identical in both cases whereas the question-type component is different.

Finding the best answer $\hat{A}$ involves a search over all A for the one which maximizes the probability of the above model:

$$\hat{A} = \arg\max_{A} P(A, l_A \mid W, X). \quad (2)$$

This is guaranteed to give us the optimal answer in a maximum likelihood sense if the probability distribution is the correct one. We don't know this and it's still difficult to model so we make various modeling assumptions to simplify things. Using Bayes' rule Equation (2) can be rearranged as:

$$\arg\max_{A} \frac{P(W, X \mid A, l_A) \cdot P(A, l_A)}{P(W, X)}. \quad (3)$$

The denominator can be ignored since it is common to all possible answer sequences and does not change. Further, to facilitate modeling we make the assumption that X is conditionally independent of W given A, that $P(X \mid A, l_A) = P(X \mid A)$, we ignore $P(X)$ and $P(W)$ since they are constant, and applying Bayes' rules we obtain the final optimisation criterion:

$$\arg\max_{A} \underbrace{P(A \mid X)}_{\substack{retrieval \\ model}} \cdot \underbrace{P(W \mid A)}_{\substack{filter \\ model}} \cdot \underbrace{P(W \mid l_A) \cdot P(l_A)}_{\substack{length \\ model}}. \quad (4)$$

The $P(A \mid X)$ model is essentially a language model which models the probability of an answer sequence A given a set of information-bearing features X. It models the proximity of A to features in X. We call this model the *retrieval model* and examine it further in Section 2.1.

The $P(W \mid A)$ model matches an answer A with features in the question-type set W. Roughly speaking this model relates ways of asking a question with classes of valid answers. For example, it associates dates, or days of the week with *when*-type questions. In general, there are many valid and equiprobable A for a given W so this component can only re-rank candidate answers retrieved by the retrieval model. If the filter model were perfect and the retrieval model were to assign the correct answer a higher probability than any other answers of the same type the correct answer should always be ranked first. Conversely, if an incorrect answer, in the same class of answers as the correct answer, is assigned a higher probability by the retrieval model we cannot recover from this error. Consequently, we call it the *filter model* and examine it further in Section 2.2.

The $P(W \mid l_A) \cdot P(l_A)$ model relates the distribution of the lengths of answers to the type of question that is being asked. For example, we might expect *Who is...* questions to be typically two words long but *How...* and *Why...* questions to be much longer. We model these probability distributions using a set of example questions together with the lengths of their answers. In tandem with the filter model, this component effectively forms a simple but effective NE tagger. We call this component the *length model* and examine it further in Section 2.3.

2.1. Retrieval model

The retrieval model essentially models the proximity of A to features in X. Since $A = a_1, \ldots, a_{l_A}$ we are actually modeling the distribution of multi-word sequences. This should be borne in mind in the following discussion whenever A is used. We currently use a deterministic information-feature mapping function $X = \mathcal{X}(Q)$. This mapping only generates word m-tuples ($m = 1, 2, \ldots$) from single words in Q that are not present in a *stop-list* of around 50 high-frequency words. function can of course extract

We assume that a corpus of text data is available for searching for answers comprising $|U|$ documents. We use the notation X_i to define an active set of the features from $x_1, \ldots, x_{l_X}$. The probability $P(A \mid X)$ is modeled as a linear interpolation of distributions:

$$P(A \mid X) = \frac{1}{Z(X)} \max\Big\{0, \sum_{X_i \in X} \lambda_{X_i} \cdot P(A \mid X_i) - \mu \cdot P(A)\Big\}.$$

where $\sum_{X_i \in X} \lambda_{X_i} = 1$, $P(A \mid X_i)$ is the conditional probability of A given the feature set X_i and μ is an answer compensation weight that de-weights the contribution of an answer according to its frequency of occurrence in the language as a whole. $Z(X)$ is the normalisation term.

2.2. Filter model

The question-type mapping function $\mathcal{W}(Q)$ extracts n-tuples ($n = 1, 2, \ldots$) of question-type features from the question Q, such as *How*, *How many* and *When were*. A set of $|V_{\mathcal{W}}| = 149$ single-word features is extracted based on frequency of occurrence in questions in previous TREC question sets. Some examples include: *when*, *where*, *who*, *whose*, *how*, *many*, *high*, *deep*, *long* etc. Modeling the complex relationship between W and A directly is non-trivial. We therefore introduce an intermediate variable representing classes of example questions-and-answers (q-and-a) c_e for $e = 1 \ldots |C_E|$ drawn from the set C_E, and to facilitate modeling we say that W is conditionally independent of c_e given A as follows:

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W, c_e \mid A) = \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot P(c_e \mid A).$$

To obtain C_E we use the Knowledge-Master questions and answers [10] and the questions and correct judgements from TREC QA evaluations prior to TREC 2002 by assigning each example q-and-a to its own unique class i.e. we do not perform clustering at all.

Making a number of other similar modeling assumptions we obtain the final formulation for the filter model:

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot \sum_{k=1}^{|C_A|} P(c_e \mid c_k) \cdot P(c_k \mid a_1) \cdots \cdots P(c_e \mid c_k) \cdot P(c_k \mid a_{l_A}). \quad (5)$$

where c_k is a concrete class in the set of $|C_A|$ answer classes C_A. Each class in C_A is ideally a homogeneous and complete set of words for a given type of question, what is usually called an answer-type in the literature, although in our model there is no explicit question- or answer-typing. For example, we expect classes of river names, mountain names, presidents' names and colors etc. The set of potential answers V_{C_A} that are clustered,

should ideally cover all words that might ever be answers to questions.

In keeping with our data-driven philosophy and related objective to make our approach as language-independent as possible we use an agglomerative clustering algorithm to derive classes automatically from data. The "seeds" for these clusters are chosen to be the most frequent $|C_A|$ words in the AQUAINT corpus. The algorithm uses the co-occurrence probabilities of words in the same corpus to group together words with similar co-occurrence statistics. For each word a in some text corpus comprising R unique words the co-occurrence probability $P(a_i \mid a, d)$ is the probability of a_i given a occurring d words away. If d is positive, a_i occurs after a, and if negative, before a. We then construct a vector of co-occurrences with maximum separation between words D, as follows:

$$\vec{a} = [P(a_1 \mid a, 1), \ldots, P(a_R \mid a, 1), P(a_1 \mid a, -1), \ldots, P(a_R \mid a, -1), \ldots\ldots, P(a_1 \mid a, D), \ldots, P(a_R \mid a, D), P(a_1 \mid a, -D), \ldots, P(a_R \mid a, -D)].$$

To find the distance between two vectors, for efficiency, we use an L_1 distance metric: $\mathcal{D}(\vec{a_i}, \vec{a_j}) = |\vec{a_i} - \vec{a_j}|$. The clustering algorithm is described in Algorithm 2.1. The obtained classes are far from perfect in terms of completeness and homogeneity but were found to give reasonable results.

Algorithm 2.1 Cluster words to generate C_A

```
initialize most frequent words to |C_A| classes
for i:= 1 to |V_{C_A}|
   for j:= 1 to |C_A|
       compute D(a_i, c_j)
   move a_i to c_j^best
   update c_j^best
```

2.3. Length model

The length model is given by $P(W \mid l_A) \cdot P(l_A)$. For the experiments in this paper we assume that the prior on the length of an answer $P(l_A)$ is uniform so applying the same approach to that in Section 2.2 we get:

$$P(W \mid l_A) = \frac{1}{Z(l_A)} \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot P(c_e \mid l_A). \quad (6)$$

3. EXPERIMENTAL WORK

In the question answering community the TREC QA task evaluations [9, 7] are the de facto standard by which QA systems are assessed and in keeping with this, we use the 500 factoid questions from the 2002 TREC QA task for system development purposes and evaluate using the 413 factoid questions in the TREC 2003 QA task.

For training the $P(W \mid A)$ model we use 288,812 example q-and-a from the Knowledge Master (KM) data [10] plus 2,408 q-and-a from the TREC-8,9 and 2001 evaluations. For evaluations on the TREC 2003 data we also add in 660 example q-and-a from the TREC 2002 development set. The most frequent $|V_{C_A}| = 224,000$ words from the AQUAINT corpus were used to obtain C_A for $|C_A| = 50, 500, 5000$ clusters using Algorithm 2.1 and $|D| = 1$. The maximum number of features used in the retrieval model was set to $l_X = 15$ for reasons of memory efficiency.

We use the Google search engine to retrieve the top $|U|$ documents for each question. The question is passed as-is to Google except that stop words from the same list described at the beginning of Section 2.1 are first removed. Text normalization is minimal and involves only removing unnecessary markup, capitalising all words and inserting sentence boundaries.

In all but one case, our evaluation is automatic and based on an exact character match between the answers provided by our system and the capitalized answers in the judgement file. We consider two sets of correctness as defined in [9, 7] where answers are either **correct** or **not exact** but where support correctness is ignored. We also look at several different metrics including the accuracy of correct answers output in the top 1, 2 and 5 answers, the number of correct and not exact answers in the top 1 position, and also at the mean reciprocal rank (MRR) and confidence-weighted score (CWS) defined in [9]. CWS ranks each answer according to the confidence that the system assigns it and gives more weight to correct answers with higher rank.

3.1. Development

Although in principle we could maximise the likelihood of each correct answer to optimise the system our final objective is the number of correct answers. Consequently we use this as our optimisation criterion on the set of 500 questions from the TREC 2002 QA task. The optimised parameters were found to be: max $l_A = 2$, $\mu = 8.0$, $|U| = 500$ and $m = n = 3$. The best set of C_A classes of those investigated was $|C_A| = 500$ classes. Table 1 gives results obtained on the development set. The 1st column shows the different evaluation metrics. The 2nd column gives the results for the best system (using the optimised parameters given above) and the 3rd column gives results using only the retrieval model.

Assessment criterion	TREC 2002 (dev. set) Best system	TREC 2002 (dev. set) Retrieval model	TREC 2003 (evaluation set) Exact	TREC 2003 (evaluation set) Human
top1 accuracy	0.326	0.208	0.232	0.276
top2 accuracy	0.386	0.270	0.310	—
top5 accuracy	0.468	0.378	0.383	—
correct	163	104	96	114
not exact	20	27	17	42
MRR	0.391	0.285	0.301	—
CWS	0.551	0.305	0.419	—

Table 1. Performance on the development set (TREC 2002) for best system and retrieval model only and performance on the evaluation set (TREC 2003) for the best development system.

3.2. Evaluation

We use an identical setup for evaluation to the best system determined on the development set except that we also include the TREC 2002 q-and-a in C_E. Two sets of results are shown in the final two columns of Table 1. The 4th column shows the results using an exact character match against the judgement file provided by TREC and the last column shows the more realistic results of a human evaluation of the same answers.

4. DISCUSSION AND ANALYSIS

Although our results on the development set are somewhat optimised they are in the mid-range of systems that participated in the TREC 2002 QA evaluation and compare very favourably with other mainly statistical systems [4, 5]. Indeed the top2 and top5 results suggest there is scope for large improvements with limited system modifications. Moreover, the CWS scores indicate that answer confidence scoring is performed particularly well by our statistical approach. On the evaluation set, however, the top1 answer performance is reduced by 29% compared to the development set performance. Below we examine where this difference in performance comes from.

First of all, it is well known there are often disagreements as to what constitutes a correct answer and TREC tends to err on the side of marking an answer correct if the data supports it. Along these lines the human-scored results in the final column of Table 1 indicate that our results are about 18% better than doing an automatic evaluation would have us believe.

In Table 2 we give the percentage of errors (i.e. incorrect answers according to the judgement file) on questions in the evaluation set that can be attributed to each model or combination of models. The analysis of these errors is necessarily subjective but is interesting nonetheless. It shows, for example, that the retrieval component has contributed by far the most errors overall.

Percentage of errors in each model combination							NOT
R	F	L	R&F	R&L	F&L	R&F&L	ERR.
29	15	11	22	2	2	9	10

Table 2. Percentage of errors in *R*etrieval, *F*ilter and *L*ength models and *NOT* actually *ERR*ors on the TREC 2003 evaluation set.

Of the data downloaded for each question from the web correct answers co-occurred in a window of 3 sentences with at least one non-stop-word query term for 81% and 79% of questions in the TREC 2002 and 2003 sets, respectively. These numbers are therefore an upper bound on the number of questions that can be answered correctly.

Using only the retrieval model we compared the performance on the development and evaluation sets. It was found that the top1, 10, 100 and 200 performance was between 11% and 25% worse on the evaluation set. Moreover, the MRR on the evaluation set was 23% worse. The filter model should boost the rank of answers with the same answer type more or less identically however if the correct answer is not given a sufficiently high rank by the retrieval model it is unlikely ever to make it into first place. Another "cheat" experiment was performed to simulate the best performance possible with our current filter model by using only the TREC 2003 QA task as our example q-and-a set C_E. The top1 accuracy was 0.390 (vs. 0.476 using the TREC 2002 QA task as C_E and testing on the development set) again corroborating what we observed above.

Finally, a quantitative analysis of the filter model was performed in two ways. The TREC 2003 questions together with all correct answers were added to the existing C_E set and all system parameters left unchanged. The top1 accuracy using this system was 0.341. Another experiment was performed in which the TREC 2003 correct answers were manually replaced with incorrect answers of the same length and same expected answer type. The top1 accuracy using this data was 0.247. This latter result is not very different from 0.232 which was the result obtained with the best development system on the evaluation data. This suggests that the $P(W \mid c_e)$ component is working well and the coverage of questions is good although the increased number of *What...* questions in the evaluation data is still poorly modeled. However, the large difference between 0.341 and 0.247 means there are serious deficiencies, most likely in the definition of C_A.

5. CONCLUSION

Although our results cannot be compared directly with other participants' in the official TREC evaluations we believe we have demonstrated that our mathematical model of question answering performs as well as many other contemporary systems on the TREC 2002 and 2003 QA tasks. Moreover, since an absolute minimum of human intervention was used to develop the system we contend that similar performance would also be obtained on other languages given enough appropriate training data examples. This is currently being confirmed on the Japanese NTCIR task.

6. ACKNOWLEDGMENTS

This research was supported by JSPS and the Japanese government 21st century COE programme. The authors also wish to thank Dietrich Klakow of Saarland University for his ideas and discussions.

7. REFERENCES

[1] Berger, A. and Caruana, R. and Cohn, D. and Freitag, D. and Mittal, V., "Bridging the Lexical Chasm: Statistical Approaches to Answer-Finding," in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, Athens, Greece, 2000.

[2] E. Brill and S. Dumais and M. Banko, "An Analysis of the AskMSR Question-answering System," in *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

[3] A. Echihabi and D. Marcu, "A Noisy-Channel Approach to Question Answering," in *Proceedings of the 41st Annual Meeting of the ACL*, 2003.

[4] A. Ittycheriah and S. Roukos, "IBM's Statistical Question Answering System—TREC-11," in *Proceedings of the TREC 2002 Conference*, 2002.

[5] Ravichandran, D. and Hovy, E. and Josef Och, F., "Statistical QA—Classifier vs. Re-ranker: What's the difference?," in *Proceedings of the ACL Workshop on Multilingual Summarization and Question Answering*, 2003.

[6] R. Soricut and E. Brill, "Automatic Question Answering: Beyond the Factoid," in *Proceedings of the HLT/NAACL 2004: Main Conference*, 2004.

[7] E. Voorhees, "Overview of the TREC 2003 Question Answering Track," in *Proceedings of the TREC 2003 Conference*, 2003.

[8] "From Factoids to Facts," The Economist, Aug. 26th 2004.

[9] E. Voorhees, "Overview of the TREC 2002 Question Answering Track," in *Proceedings of the TREC 2002 Conference*, 2002.

[10] Academic Hallmarks, "Knowledge Master Educational Software," PO Box 998, Durango, CO 81302 http://www.greatauk.com/, 2002.