

A Statistical Classification Approach to Question Answering using Web Data

Edward Whittaker 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

Dietrich Klakow

Lehrstuhl für Sprachsignalverarbeitung

Saarland University

D-66041 Saarbrücken

Germany

dietrich.klakow@lsv.uni-saarland.de

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 33% of questions are answered correctly. Performance is then evaluated on the factoid questions of the TREC 2003 QA task where 23% of questions were answered correctly, which would rank the system in the top 10 of contemporary QA systems on the same task.

1. Introduction

Question answering (QA), particularly in the style of TREC, 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 [2, 3, 6, 9, 15, 16, 17]. In this paper, we present a new, general framework for QA and evaluate its performance on the TREC 2003 QA task [19].

QA, especially in the context of the Web, has been cited recently as the next-big-thing in search technology [1]. Exploitation of the huge domain coverage and redundancy inherent in web data has also become a theme in TREC participants’ systems [3, 4, 10, 15, 17, 20]. 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 [18] 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 QA. In [2] a number of statistical methods is investigated for what was termed bridging the lexical gap between questions and answers. In [9] a maximum-entropy based classifier using several different features was used to determine answer correctness and in [16] performance was compared against classifying the actual answer. A statistical noisy-channel model was used in [6] in which the distance computation between query and candidate answer sentences is performed in the space of parse trees. In [17] the lexical gap is bridged using a statistical translation model. Of these, our approach is probably most similar to [17] and the re-ranker in [16]. Statistical approaches still under-perform the best TREC systems e.g. [11] 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 [8, 11, 12, 14], named-entity (NE) extraction, or any other linguistic information e.g from semantic analysis [8] or from question parsing [8, 9, 11]. 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.

The paper is organized as follows: we first derive a mathematical framework for QA as a classification task in Section 2, then give experimental results in Section 3. A discussion, further work and conclusion are given in Sections 4 and 5.

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 $A = a_1, \dots, a_{l_A}$ and l_Q words $Q = q_1, \dots, q_{l_Q}$, 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:

$$P(A | Q) = P(A, l_A | W, X), \quad (1)$$

where $W = w_1, \dots, w_{l_W}$ 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 $X = x_1, \dots, x_{l_X}$ 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:

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

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 this can be rearranged as

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

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 to obtain:

$$\arg \max_A P(X | A, l_A) \cdot P(W | A, l_A) \cdot P(A, l_A). \quad (4)$$

Using Bayes rule, making further conditional independence assumptions and assuming uniform prior probabilities, which therefore do not affect the optimisation criterion, we obtain the final optimisation criterion:

$$\arg \max_A \underbrace{P(A | X)}_{\text{retrieval model}} \cdot \underbrace{P(W | A)}_{\text{filter model}} \cdot \underbrace{P(W | l_A)}_{\text{length model}}. \quad (5)$$

The $P(A | X)$ model is essentially a language model which models the probability of an answer sequence A given a set of information-bearing features X , similar to the work of [13]. 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 $P(W | A)$ 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 $P(W | l_A)$ 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 $A = a_1, \dots, a_{l_A}$ we are actually modeling the distribution of multi-word sequences.

This should be borne in mind in the following discussion whenever A is used. As mentioned above, we currently use a deterministic information-feature mapping function $X = \mathcal{X}(Q)$. This mapping only generates word m -tuples ($m = 1, 2, \dots$) from single words in Q that are not present in a *stop-list* of around 50 high-frequency words. In principle the function can of course extract deeper linguistic features but we leave this for future work.

We first assume that a corpus of text data S is available for searching for answers comprising $|S|$ sentences $S_1, \dots, S_{|S|}$ and $|U|$ documents and a vocabulary V of $|V|$ unique words. We use the notation X_i to define an active set of the features x_1, \dots, x_{l_x} such that $X_i = x_1 \cdot \delta(d_1), x_2 \cdot \delta(d_2), \dots, x_{l_x} \cdot \delta(d_{l_x})$ where $\delta(\cdot)$ is a discrete indicator function which equals 1 if its argument evaluates true (i.e. its argument(s) are equal, is not an empty set, or is a positive number) and 0 if false (i.e. its argument(s) are not equal, is an empty set, is 0 or is a negative number) and $\vec{d} = [d_1, \dots, d_{l_x}]$ is the solution¹ to $i = \sum_{j=1}^{l_x} 2^{j-1} d_j$.

The probability $P(A | X)$ is modeled as a linear interpolation of the 2^{l_x} distributions²:

$$P(A | X) = \sum_{i=1}^{2^{l_x}} \lambda_{X_i} \cdot P(A | X_i), \quad (6)$$

where $\lambda_{X_i} = 1/2^{l_x}$ for all i and $P(A | X_i)$ is the conditional probability of A given the feature set X_i and is computed as the maximum likelihood estimate from the corpus S .

2.2. Filter model

The question-type mapping function $\mathcal{W}(Q)$ extracts n -tuples ($n = 1, 2, \dots$) 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 $e = 1 \dots |C_E|$ drawn from the set C_E , and to facilitate modeling we say that W is conditionally independent of c_e given A as follows:

¹Note that the value of i is simply the base10 number that represents the binary encoding of the active features in X_i .

²A linear interpolation of models, which borrows directly from statistical language modeling techniques for speech recognition, was found to give retrieval performance approximately twice that of a naive-Bayes or log-linear formulation.

$$\begin{aligned} P(W | A) &= \sum_{e=1}^{|C_E|} P(W, c_e | A) \\ &= \sum_{e=1}^{|C_E|} P(W | c_e) \cdot P(c_e | A). \end{aligned} \quad (7)$$

Given a set E of example q-and-a t_j for $j = 1 \dots |E|$ where $t_j = (q_1^j, \dots, q_{l_Q}^j, a_1^j, \dots, a_{l_A}^j)$ we define a mapping function $f : E \mapsto C_E$ by $f(t_j) = e$. Each class $c_e = (w_1^e, \dots, w_{l_W}^e, a_1^e, \dots, a_{l_A}^e)$ is then obtained by $c_e = \bigcup_{j:f(t_j)=e} \mathcal{W}(t_j) \bigcup_{i=1}^{l_A} a_i^j$, so that:

$$P(W | A) = \sum_{e=1}^{|C_E|} P(W | w_1^e, \dots, w_{l_W}^e) \cdot P(a_1^e, \dots, a_{l_A}^e | A).$$

Assuming conditional independence of the answer words in class c_e given A , and making the modelling assumption that the j th answer word a_j^e in the example class c_e is dependent only on the j th answer word in A we obtain:

$$P(W | A) = \sum_{e=1}^{|C_E|} P(W | c_e) \cdot \prod_{j=1}^{l_A} P(a_j^e | a_j).$$

Since our set of example q-and-a cannot be expected to cover all the possible answers to questions that may be asked we perform a similar operation to that above to give us the following:

$$P(W | A) = \sum_{e=1}^{|C_E|} P(W | c_e) \prod_{j=1}^{l_A} \sum_{a=1}^{|C_A|} P(a_j^e | c_a) P(c_a | a_j) \quad (8)$$

where c_a is a concrete class in the set of $|C_A|$ answer classes C_A . The independence assumption leads to underestimating the probabilities of multi-word answers so we take the geometric mean of the length of the answer (not shown in Equation (8)) and normalize $P(W | A)$ accordingly.

2.2.1 Obtaining C_E

As mentioned above, the model given by Equation (8) envisaged using clusters of example q-and-a. There are two difficulties with such an approach. The first regards the

availability of suitable training data and the second concerns the space in which comparisons should be made to obtain good clusters.

The first problem is easily surmounted, for English at any rate, with the availability of large collections of example q-and-a. We use the Knowledge-Master (KM) questions and answers [7] and the questions and correct judgments from TREC QA evaluations prior to TREC 2002.

The second problem was solved by assigning each example q-and-a to its own unique class and not performing clustering at all. Some techniques for producing robust and coherent classes were investigated but without much success.

2.2.2 Obtaining C_A

The classes C_A are intended to be expansions of the answer parts of the example q-and-a in C_E and as such they are inextricably related. 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, colors, ways of dying etc. Clearly, an overlap between classes is expected and is desirable thus words can belong to any number of classes, though only occur at most once in any given class.

Moreover, V_{C_A} , the set of potential answers that are clustered, should ideally cover all possible words that might ever be answers to questions. For many potential answers we can't satisfy this since new words enter usage periodically and even covering all possible current words is impossible. However, we can take the most frequent V_{C_A} words from some corpus as an approximation and could also cover many other potential answers implicitly e.g. dates, currency amounts, numbers etc. with the use of regular expressions.

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 $P(a_i | a, d)$ 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:

$$\vec{a} = [P(a_1 | a, 1), \dots, P(a_R | a, 1), P(a_1 | a, -1), \dots, P(a_R | a, -1), \dots, P(a_1 | a, D), \dots, P(a_R | a, D), P(a_1 | a, -D), \dots, P(a_R | a, -D)]. \quad (9)$$

Rather than storing $2DR^2$ elements we can compute most terms efficiently and on-the-fly using the Katz back-off method and absolute discounting for estimating the probabilities of unobserved events. To find the distance between two vectors, for efficiency, we use an L_1 distance metric: $D(\vec{a}_i, \vec{a}_j) = |\vec{a}_i - \vec{a}_j|$. Merging two vectors then involves a straightforward update of the co-occurrence counts for a cluster and re-computing the affected conditional probabilities and back-off weights. The clustering algorithm is described below:

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  $D(\vec{a}_i, \vec{c}_j)$ 
  move  $\vec{a}_i$  to  $c_j^{best}$ 
  update  $c_j^{best}$ 

```

The classes that are obtained are far from perfect in terms of completeness and homogeneity but were found to give satisfactory performance.

2.3. Length model

The length model is given by $P(W | l_A)$. Applying the same approach to that in Section 2.2 we get:

$$P(W | l_A) = \frac{1}{Z(l_A)} \sum_{e=1}^{|C_E|} P(W | c_e) \cdot P(c_e | l_A). \quad (10)$$

The $P(W | c_e)$ distribution is modeled the same as for Equation (8) and

$$P(c_e | l_A) = \frac{\delta(l_{A^e} = l_A)}{\sum_{e'=1}^{|C_E|} \delta(l_{A^{e'}} = l_A)}. \quad (11)$$

The $P(W | l_A)$ distribution is then normalised using $Z(l_A) = \sum_W P(W | l_A)$ to ensure it is a correct probability distribution. Some examples of the length probability distributions of answers for different questions are shown in Figure 1.

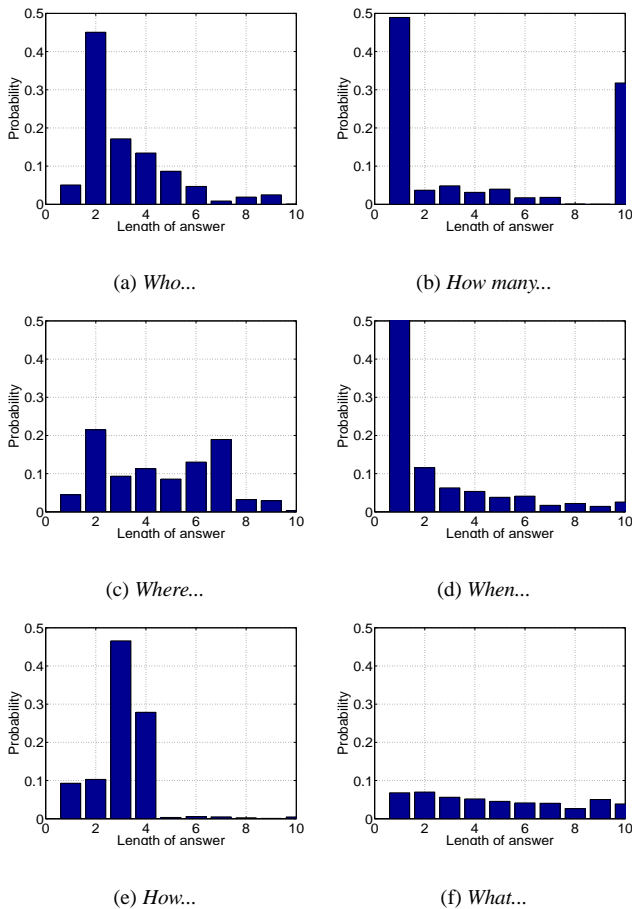


Figure 1. Distribution of length probabilities predicted by the length model for 6 types of question.

3. Experimental Work

In the QA community the TREC QA task evaluations [18, 19] 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 $P(W | A)$ model we use 288,812 example q-and-a from the Knowledge Master KM) data [7] plus 2,408 q-and-a from the TREC-8,9 and 2001 evaluations ($|C_E| = |E| = 291, 220$). For evaluations on the TREC 2003 data we also add in 660 example q-and-a from the TREC 2002 development set ($|C_E| = |E| = 291, 880$).

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. Care is taken to remove any pages referring to the TREC evalua-

tions which often cite examples of questions together with the answers. Text processing is minimal and involves only removing unnecessary markup, capitalising all words and inserting sentence boundaries.

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 2.2 and $|D| = 1$. The vocabulary V_{C_A} covers approximately 90% of the answers in E . The maximum number of features used in the retrieval model was set to $l_X = 15$ for reasons of speed and memory efficiency.

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 [18, 19] where answers are either **correct** or **not exact** but where support correctness is ignored³. 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) [18] defined below:

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^N \frac{1}{\text{rank of correct answer}} \quad (12)$$

$$\text{CWS} = \frac{1}{N} \sum_{i=1}^N \frac{\# \text{ correct upto question } i}{i} \quad (13)$$

where N is the number of questions and where for the CWS metric answers are sorted by the confidence score assigned to an answer.

3.1. Development: TREC 2002

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 best set of C_A classes of those investigated was $|C_A| = 500$ classes. Performance against $|U|$, the maximum number of documents downloaded from Google for each question is shown in Figure 2 for two different classifications of correctness. Of the values investigated $|U| = 500$ was found to give the best results. Interestingly, these results are rather different to those reported in [5] where an optimal number of “snippets” (summaries of relevant parts of different web-pages) of around 200 is observed. However, there are significant differences between our approach and theirs: we use the

³The current system does not output **NIL** when an answer cannot be found so we automatically get all such answers wrong.

Assessment criterion	System description/performance on TREC 2002						TREC 2003 (evaluation set)	
	Best system	Retrieval model only	$ C_A $		No KM data	AQUINT corpus only	Exact	Human
			50	5000				
top1 accuracy	0.326	0.208	0.304	0.292	0.280	0.156	0.232	0.276
top2 accuracy	0.386	0.270	0.378	0.372	0.338	0.192	0.310	—
top5 accuracy	0.468	0.378	0.442	0.458	0.436	0.232	0.383	—
correct	163	104	152	146	140	78	96	114
not exact	20	27	20	19	27	5	17	42
MRR	0.391	0.285	0.371	0.368	0.350	0.194	0.301	—
CWS	0.551	0.305	0.538	0.488	0.470	0.322	0.419	—

Table 1. Performance on the development set (TREC 2002) for best system and systems that differ by one parameter, and performance on the evaluation set (TREC 2003) for the best development system.

entire document rather than the snippet; and, we only use a query in which all question terms are ANDed together as opposed to their use of several, weighted query re-writes.

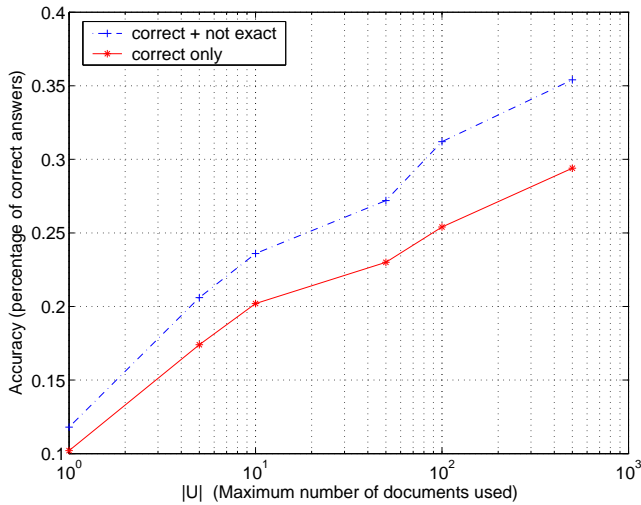


Figure 2. System accuracy for $\max l_A = 1$ vs. $|U|$ (the maximum number of documents used).

Columns 2 to 7 in Table 1 give the 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). To show the effect of different system components on performance the subsequent columns give results using different system configurations. The 3rd column gives results for the retrieval model only i.e. no filter or length model. The 4th and 5th columns show results for two different values of $|C_A|$. The 6th column is when only the TREC8, 9 and 2001 example q-and-a are used i.e. no KM data. The 7th column uses only the

AQUINT corpus to find answers with no use of any web data.

3.2. Evaluation: TREC 2003

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 last but one column shows the results using an exact character match against the judgement file provided by NIST and the last column shows the more realistic results of a human evaluation of the same answers.

In Table 2 we give a breakdown of the time taken to answer a question, using a 3.2GHz Pentium4 processor running Linux with 2Gb RAM and a Gigabit internet connection, averaged over the 413 questions of the evaluation set.

System component	Cumulative response time
Document download	42 secs
+Retrieval model	56 secs
+Filter model	74 secs

Table 2. Timings for system components averaged over the 413 questions of the evaluation set.

4. Discussion and analysis

Although our results on the development set are somewhat optimised they would place us in the top ten of systems that participated in the TREC 2002 QA evaluation and compare very favourably with other mainly statistical systems [9, 16]. 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 (TREC 2003), however, the top1 answer performance is reduced by 29% compared to the development set performance. In this section we examine where this difference in performance comes from.

In Table 3 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 errors in each model combination (%)							NOT ERR.
R	F	L	R&F	R&L	F&L	R&F&L	
29	15	11	22	2	2	9	10

Table 3. Percentage of errors in Retrieval, Filter and Length models and NOT actually ERRors 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 an upper bound on the number of questions that can be correctly answered and indicate that most question’s data contained correct answers although we don’t know anything about the *signal-to-noise* ratios.

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 experiment was performed to simulate the best performance we could ever expect to obtain 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. Exactly why the retrieval model performs worse on the evaluation set will be examined in future work.

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 per-

formed 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 $P(W | c_e)$ component is working well and the coverage of E is good although the increased number of *What...* questions in the evaluation data is still poorly modeled as indicated by the distributions of length probabilities in Figure 1(f). However, the large difference between 0.341 and 0.247 means there are serious deficiencies in either the modeling of $P(c_a | a)$ or in the definition of C_A . We expect the latter to be most likely and this will be a focus of future work.

It was found that the length model brought no benefit to the overall performance since the optimal value of β was 0. The most likely explanation is that the same information is already encoded in the filter model since the class membership of candidate answers is dependent on the length of an example answer and the position of the class within that example answer. In addition, we take the geometric mean of the score which also factors out the contribution of length to some extent.

It is well known there are often disagreements as to what constitutes a correct answer and NIST typically marks 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.

5. Conclusion and future work

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.

Future work will focus on solving the deficiencies in the retrieval model such as improving weighting of the individual distributions and the selection of robust features. We will also look at alternatives to modeling $P(W | A)$ such as using SVD on a wide range of (both surface and deeper linguistic) features and performing the distance computation between example questions and the query in a reduced space. We will also examine whether we can avoid clustering C_A and instead model $P(c_e | A)$ directly. Since the statistical approach works well in the presence of repeated instances of the correct answer endeavours will also

be made to improve data collection through standard TREC approaches such as chunking, question re-writes and using reliable information sources such as encyclopaediae. It is relatively easy to extend our approach to answering list questions but using our statistical approach also opens up the interesting possibility of generating consensus answers to definition questions. Both will be examined in future work.

6. Acknowledgments

This research was supported by JSPS and the Japanese government 21st century COE programme.

References

- [1] From Factoids to Facts. *The Economist*, Aug. 26th 2004.
- [2] A. Berger, R. Caruana, D. Cohn, D. Freitag, and V. Mittal. 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.
- [3] E. Brill, 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.
- [4] C. L. A. Clarke, G. V. Cormack, and T. R. Lynam. Exploiting Redundancy in Question Answering. In *Proceedings of the 24th Annual International ACM SIGIR Conference on research and Development in Information Retrieval*, New Orleans, 2001.
- [5] S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng. Web Question Answering: is more always better? In *Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval*, Tampere, Finland, 2002.
- [6] A. Echihabi and D. Marcu. A Noisy-Channel Approach to Question Answering. In *Proceedings of the 41st Annual Meeting of the ACL*, 2003.
- [7] A. Hallmarks. Knowledge Master Educational Software. PO Box 998, Durango, CO 81302 <http://www.greatauk.com/>, 2002.
- [8] E. Hovy, U. Hermjakob, and L. C-Y. The Use of External Knowledge in Factoid QA. In *Proceedings of the TREC 2001 Conference*, 2001.
- [9] A. Ittycheriah and S. Roukos. IBM's Statistical Question Answering System—TREC-11. In *Proceedings of the TREC 2002 Conference*, 2002.
- [10] C. Kwok, O. Etzioni, and D. Weld. Scaling Question Answering to the Web. In *ACM Transactions of Information Systems*, 2001.
- [11] D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Laccatusu, A. Novischi, A. Badulescu, and O. Bolohan. LCC Tools for Question Answering. In *Proceedings of the TREC 2002 Conference*, 2002.
- [12] M. Pasca and S. Harabagiu. The Informative Role of WordNet in Open-Domain Question Answering. In *Proceedings of the NAACL 2001 Workshop on WordNet and Other Lexical Resources*, Pittsburgh PA, 2001.
- [13] J. Ponte and W. Croft. A Language Modeling Approach to Information Retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval*, Melbourne, Australia, 1998.
- [14] J. Prager, J. Chu-Carroll, and K. Czuba. Use of WordNet Hypernyms for Answering What-Is Questions. In *Proceedings of the TREC 2002 Conference*, 2002.
- [15] D. Radev, W. Fan, H. Qi, H. Wu, and A. Grewal. Probabilistic Question Answering on the Web. In *Proc. of the 11th international conference on WWW*, Hawaii, US, 2002.
- [16] D. Ravichandran, E. Hovy, and F. Josef Och. Statistical QA—Classifier vs. Re-ranker: What's the difference? In *Proceedings of the ACL Workshop on Multilingual Summarization and Question Answering*, 2003.
- [17] R. Soricut and E. Brill. Automatic Question Answering: Beyond the Factoid. In *Proceedings of the HLT/NAACL 2004: Main Conference*, 2004.
- [18] E. Voorhees. Overview of the TREC 2002 Question Answering Track. In *Proceedings of the TREC 2002 Conference*, 2002.
- [19] E. Voorhees. Overview of the TREC 2003 Question Answering Track. In *Proceedings of the TREC 2003 Conference*, 2003.
- [20] J. Xu, A. Licuanan, J. May, S. Miller, and R. Weischedel. TREC 2002 QA at BBN: Answer Selection and Confidence Estimation. In *Proceedings of the TREC 2002 Conference*, 2002.