TREC2005 Question Answering Experiments at Tokyo Institute of Technology

Edward Whittaker Pierre Chatain Sadaoki Furui
Dept. of Computer Science
Tokyo Institute of Technology
2-12-1, Ookayama, Meguro-ku
Tokyo 152-8552 Japan
{edw,pierre,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 describe Tokyo Institute of Technology's speech group's first attempt at the TREC2005 question answering track which placed us eleventh overall among the best systems of the 30 participants in the track. All our evaluation systems were based on novel, non-linguistic, data-driven approaches to question answering. Our main focus was on the factoid task and we describe in detail one of the new models used in this year's evaluation runs. The list task was treated as a simple extension of the factoid task while the other question task was treated as an automatic summarization problem by important sentence selection. Our best system on the factoid task gave 21.3% correct in first place; our best result on the list task was an average F-score of 0.069 and on the other question task a best average F-score of 0.138.

1 Introduction

In this paper, we describe the application of a new, general, data-driven and non-linguistic framework for the factoid task of TREC2005 that was presented previously in [18]. We believe our approach is substantially different to conventional approaches though it shares elements of other statistical, data-driven approaches to factoid question answering in the literature [1, 2, 4, 7, 15, 16, 17].

The availability of large amounts of data, both for system training and answer extraction logically leads to examining statistical approaches to QA. In [1] a number of statistical methods is investigated for what was termed bridging the lexical gap between questions and answers. In [7] 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 [4] 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.

The system we developed for the factoid QA task in TREC2005 involves a statistical, noisy-channel approach where we treat QA as a classification problem. We use a new mathematical model that can include all kinds of dependencies in a consistent manner and is fully trainable requiring minimal human intervention once sufficient data is collected. 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 and to remove the need for linguistic modules that require a lot of effort to create.

There are several major differences between our approach and most contemporary approaches to QA: for example, we only use capitalised word tokens in our system and do not use WordNet [6, 11, 12, 14], named-entity (NE) extraction, or any other linguistic information e.g. from semantic analysis [6] or from question parsing [6, 7, 11]. We also rely heavily on the web and a conventional web search engine as a source of data for answering questions[1]. We also want to make clear that our approach is also very different to other purely web-based approaches such as askMSR [2] and Aranea [10]. For example, we use entire documents rather than the snippets of text returned by web search engines; we do not use structured document sources or databases and we do not transform the query in any way either by term re-ordering or by modifying the tense of verbs.

Three runs were submitted (`asked05a,b,c`) for evaluation, all of which were based on variations of this new statistical approach. For the list task, an extension to the system used in the factoid task is used. For the *other* question task a variation on a system used for speech summarization [8] is employed.

The rest of the paper is organized as follows: we first present a summary in Section 2 of the mathematical framework for factoid QA as a classification task that was presented in [18]. We then describe the extension of our factoid QA approach to answering list questions in Section 3 and the automatic summarization approach applied to *other* questions in Section 4. We then describe our experimental setup and the performance on all 3 tasks of TREC2005 in Section 7. A discussion and conclusion are given in Sections 8 and 9.

2 Factoid question 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 offline 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 depends on two sets of features $W = \mathcal{W}(Q)$ and $X = \mathcal{X}(Q)$ as follows:

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

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:

$$\hat{A} = \arg\max_A P(A \mid W, X). \tag{2}$$

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

$$\arg\max_A \frac{P(W, X \mid A) \cdot P(A)}{P(W, X)}. \tag{3}$$

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:

$$\arg\max_A P(X \mid A) \cdot P(W \mid A) \cdot P(A). \tag{4}$$

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:

$$\arg\max_A \underbrace{P(A \mid X)}_{\substack{retrieval\\ model}} \cdot \underbrace{P(W \mid A)}_{\substack{filter\\ model}}. \tag{5}$$

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, 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 $P(W \mid A)$ 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

[1]For interest's sake, however, in this year's TREC we also performed one run that used no web data at all.

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.

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. As mentioned above, we currently use a deterministic information-feature mapping function $X = \mathcal{X}(Q)$. This mapping only generates word m-tuples ($m = 1, 2, \ldots$) from single words in Q that are not present in a *stop-list* of around 50 high-frequency words. In principle the function could 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 $S_1, \ldots, S_{|S|}$ 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 $x_1, \ldots, x_{l_X}$ such that $X_i = x_1 \cdot \delta(d_1), x_2 \cdot \delta(d_2), \ldots, x_{l_X} \cdot \delta(d_{l_X})$ 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 $\vec{d} = [d_1, \ldots, d_{l_X}]$ is the solution[2] to $i = \sum_{j=1}^{l_X} 2^{j-1} d_j$.

The probability $P(A \mid X)$ is modeled as a linear interpolation of the 2^{l_X} distributions[3]:

$$P(A \mid X) = \sum_{i=0}^{2^{l_X - 1}} \lambda_{X_i} \cdot P(A \mid X_i), \tag{6}$$

where $\lambda_{X_i} = 1/2^{l_X}$ for all i, $P(A \mid X_0)$ is a zerogram distribution, and $P(A \mid X_i)$ is the conditional probability of A given the feature set X_i and is computed as the maximum likelihood estimate from the corpus S:

$$P(A \mid X_i) = \frac{N(A, X_i)}{N(X_i)}, \tag{7}$$

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

[3]A 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.

where

$$N(A, X_i) = \sum_{j=1}^{|S|} \delta(X_i \in \mathcal{X}(S_j)) \cdot \delta(A \in S_j), \tag{8}$$

$$N(X_i) = \sum_{v \in V} N(v, X_i). \tag{9}$$

We modify Equation (8) to include contributions from adjacent sentences weighted by λ_{adj} which typically has a value ≤ 1:

$$N(A, X_i) = \sum_{j=1}^{|S|} \delta(X_i \in \mathcal{X}(S_j)) \cdot \max\{\delta(A \in S_j), \lambda_{adj} \cdot \delta(A \in S_{j-1}), \lambda_{adj} \cdot \delta(A \in S_{j+1})\}. \tag{10}$$

It turns out that smoothing the maximum likelihood estimates from each component distribution has little effect on performance so none is performed. This is partly because of the inherent smoothing effect achieved by interpolating all the distributions together and partly since there is no need to smooth for non-occurring events since such zerotons are never likely to be selected as answers.

One clear deficiency, however, is the use of equal-valued interpolation weights for all distributions. One might expect a dependence on the number of active features or on $N(X_i)$, however, no such reliable relationship has so far been determined although investigations continue.

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 $|V_{\mathcal{W}}| = 2522$ 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:

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W, c_e \mid A) \tag{11}$$

$$= \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot P(c_e \mid A). \tag{12}$$

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

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W \mid w_1^e, \ldots, w_{l_{W^e}}^e) \cdot P(a_1^e, \ldots, a_{l_{A^e}}^e \mid A). \quad (13)$$

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

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot \prod_{j=1}^{l_{A^e}} P(a_j^e \mid a_j). \quad (14)$$

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:

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W \mid c_e) \prod_{j=1}^{l_{A^e}} \sum_{a=1}^{|C_A|} P(a_j^e \mid c_a) P(c_a \mid a_j), \quad (15)$$

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 (15)) and normalize $P(W \mid A)$ accordingly.

The system using the above formulation of filter model given by Equation (15) is referred to as model ONE. Systems using the model given by Equation (13) are referred to as model TWO. The training of Model ONE has been described in detail in [18]. The details of Model TWO will be described in a future publication.

2.3 Reconciling $P(A \mid X)$ and $P(W \mid A)$

The approach to QA that has been presented is similar in essence to that of approaches to automatic speech recognition (ASR) where there are separate acoustic and language models. In ASR, it is necessary to include a *language model weight*, α, which raises the probabilities given by the language model to the power α, otherwise performance is very poor:

$$\hat{A} = \arg\max_A \frac{P(A \mid X)^\alpha \cdot P(W \mid A)}{\sum_{A'} P(A' \mid X)^\alpha \cdot P(W \mid A')}.$$

Several, possibly related, explanations have been given for this requirement including compensation for the independence assumption. In any case, the dynamic range of the models is typically very different and needs compensating somehow. α can be optimised easily once the individual models have been optimised separately.

3 List question task

For the list task we essentially use identical systems to those used in the factoid task. Our factoid QA systems always output a list of all the possible answers they encounter in the data, ranked by their probabilities. The issue for the list task is therefore to determine how many of the top answers to output so as to maximise the F-score. We investigated different methods during the development phase for selecting output thresholds. These are discussed for each of the three different runs we submitted in Section 7.4.

4 Other question task

We treat the answering of *other* questions as a summarization task and employ a variation on a method used for speech summarization [8] for this purpose. The data from which the nuggets are to be extracted (either web or AQUAINT) is first cleaned to remove words that are unlikely to be required in a nugget but which occur frequently in the data. Duplicate sentences are also removed along with sentences shorter than 40 bytes and longer than 220 bytes. We then select up to 500 sentences which contain as many of the topic words associated with the question as possible, assigning a score to each topic word based on an idf value obtained from the AQUAINT corpus. This results in a single document which is then summarized by selecting up to 175 important sentences according to a combination of a linguistic score (using a 3-gram language model) and a significance score (measured by a tf/idf score), according to the following:

$$S(W) = \frac{1}{N} \sum_{i=1}^{N} \{L(w_i) + \alpha \cdot I(w_i)\}, \quad (16)$$

where N is the number of words in the sentence W, and $L(w_i)$ and $I(w_i)$ are the linguistic score and the significance score of word w_i, respectively. Sentences over 140

System	Target data source	Which model	Submitted run
`asked05a`	AQUAINT	ONE	yes
`asked05b`	Web	TWO	yes
`asked05c`	AQUAINT+Web	ONE+TWO	yes
`asked05d`	Web	ONE	no

Table 1. Descriptions of systems developed for TREC2005.

bytes are compacted so that all nuggets have a length between 40 and 140 bytes, using a similar summarization process. Finally, upto NU_{max} nuggets are selected according to their final summarization score, making sure that the byte-wise Levenstein distance between two nuggets is less than $R\%$ of the bytes in any previously selected sentence. Once the set of nuggets had been determined no attempt was made to suppress nuggets that contained answers already given for factoid or list questions.

5 System combination

For one run this year, for all 3 tasks we combined the output from 3 different systems and submitted this as a separate run. For the factoid and list tasks this combination is performed by summing the inverse rank of an answer a from each component system s to generate a new score for the answer as follows:

$$score(a) = \sum_{s} \frac{1}{r_s(a)}, \tag{17}$$

where $r_s(a)$ is the rank of answer a in system s. If a is not output by system s we define $r_s(a) = \infty$. The answers, sorted by their new score, then form the ranked output of the combined system.

For the *other* question task, system combination was performed simply by concatenating nuggets from two systems upto a maximum number NU_{max} of nuggets.

6 Support generation

The Aranea system [10] was fortuitously released a few months prior to the TREC2005 evaluation and we took the code for the *ProjectAnswer* module and made a few simple changes to suit the kind of answers we needed to search for (e.g. all upper-cased answers in all upper-cased text). In all cases, only the (upto) 1000 documents retrieved by the PRISE search engine and provided by NIST were used for searching for support information for each question (i.e. not the documents retrieved by our system for the document-ranking task). The same tool was used for determining support for answers in all 3 tasks.

7 Experimental work

Three different systems (`asked05a,b,c`) were submitted for evaluation with characteristics given in Table 1. System `asked05a` uses model ONE and only AQUAINT data. System `asked05b` uses model TWO and only Web data. System `asked05d` uses model ONE and only Web data. System `asked05c` is a combination of the outputs from systems `asked05a`, `asked05b` and `asked05d` combined according to the method presented in Section 5.

7.1 Question pre-processing

Conversion from the XML format provided by NIST to that required by our system was elementary. For each question set the target is extracted and each component question extracted. All target and question strings are then mapped to upper-case. All punctuation except for "'S" is removed both from target and question strings (for some reason commas were not removed but this did not cause any problems). Then, if the target for a question does not appear character-for-character in that question string it is simply appended to the end of the question string. In general, we feel our approach is quite robust to errors in pre-processing so we do not worry too much about it.

In addition, although the questions in each set are supposed to be part of a dialogue in which subsequent questions can reference prior questions and answers in the same set, we do not attempt to exploit this. Consequently, each question is treated independently of all other questions.

7.2 Target document preparation

Our system was designed with web-based question answering in mind. However, for the sake of interest we also performed one run (`asked05a`) which only used the (upto) 1000 documents from the AQUAINT corpus retrieved by the PRISE search engine and supplied by NIST. The other source of documents we used was obtained by passing each pre-processed, upper-cased question as-is to a web search engine; the top 500 text or HTML documents returned were then downloaded and kept separate for each question. (We relied on the web search engine to strip out stop words from

System	Factoid task			List task	Other task	Avg. per-series score
	Right	Unsupp.	ineXact			
`asked05a`	45 (12.4%)	7 (1.9%)	21 (5.8%)	0.044	0.138	0.108
`asked05b`	72 (19.9%)	19 (5.2%)	21 (5.8%)	0.057	0.091	0.136
`asked05c`	77 (21.3%)	19 (5.2%)	22 (6.1%)	0.069	0.131	0.157
`asked05d`	64 (17.7%)		10 (2.8%)	—	—	—

Table 2. Performance on all 3 tasks of the 3 submitted runs and an estimated performance score for the factoid task of run `asked05d` which was not submitted for evaluation.

the query.) In contrast to other experiments using web data in the literature [3] none of our experiments has yet found a point at which performance deteriorates after a certain number of documents. We therefore settled on 500 documents for reasons of expediency rather than optimality. Subsequent text processing of the downloaded documents proceeds in essentially the same way as for question preprocessing except that HTML markup is also removed and sentence boundaries are inserted.

7.3 Factoid question task

For system development we optimise performance on the TREC2002,3 and 4 evaluation questions using a rotating form of cross-validation but with an emphasis on the performance on the TREC2004 questions. For training the filter model we use 288812 example q-and-a from the Knowledge Master KM data [5] plus 2408 q-and-a from the TREC-8,9 and TREC2001 questions, and also the TREC2002,3,4 evaluation q-and-a in a rotating manner so as not to include test questions as examples during development.

The most frequent $|V_{C_A}| = 224000$ words from the AQUAINT corpus were used to obtain C_A for $|C_A| = 50, 500, 5000$ clusters as described in [18]. 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.

Answer accuracy for the TREC2002,3 and 4 test sets is computed automatically and is based on an exact character match between the answers provided by our system and the capitalized answers in the judgment files provided by NIST. For development we do not worry about support information assuming that this can be constructed reliably later on. Also, the current system never outputs NIL when an answer cannot be found so we automatically get all such answers wrong in both development and evaluation.

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 1341 questions from the TREC2002,3 and 4 QA tasks. The optimised parameters were found to be: $m = 3$, $n = 3$, $\lambda_{adj} = 0.3$, $|U| = 500$, and $\alpha = 2.0$. The best set of C_A classes of those investigated was $|C_A| = 5000$ classes[4].

For our evaluation system we use an identical setup to the best system determined during development except that we included the TREC2002,3 and 4 q-and-a permanently in C_E ($|C_E| = |E| = 291220$). The results for all 3 runs on all 3 tasks are shown in Table 2 together with an estimated performance for run `asked05d` which was not submitted for evaluation.

7.4 List question task

System development proceeded in a manner essentially identical to that for the factoid question task described in the previous section, except that the list q-and-a from TREC2002,3 and 4 were also used and added to the set of example q-and-a using the rotating method of cross-validation.

For the evaluation system the best system determined during development was selected with the following parameter settings: $m = 3$, $n = 3$, $\lambda_{adj} = 0.3$, $|U| = 500$, $\alpha = 0.5$ and $|C_A| = 5000$ classes. In addition the list q-and-a from TREC2002,3 and 4 were permanently added to C_E.

The number of questions to output was different for each of the 3 runs we submitted and was determined during development under conditions expected to be similar to the evaluation conditions in each case. For run `asked05a` we selected 5 answers and then performed answer filtering which typically resulted in fewer than 5 answers per question. For runs `asked05b,d` we performed answer filtering first and then selected the top 5 answers. For run `asked05c` (using system combination) we simply used the set of answers from runs `asked05a,b,d` which resulted in between 11 and 15 answers per question.

[4] There may be a more optimal number or combination of such classes.

7.5 Other question task

System development for the *other* question task for all 3 runs was performed using only the TREC2004 *other* questions and evaluated using POURPRE-1.0c [9] with the metric based on simple term counts. During development we determined the optimal number of nuggets to output for runs `asked05a,b,c` as $NU_{max} = 16, 19$ and 18, respectively, the length of nuggets produced by our system to be between 40 and 140 bytes and set $R = 83\%$.

8 Discussion and analysis

Our best run, run `asked05c`, ranked 11th among the best systems from each of the 30 participants on the factoid question task this year. While the performance of run `asked05b` was almost as good as and contributed most to the performance of run `asked05c`, the performance of run `asked05a` was quite low, as expected from our development experiments. For the analysis of our system performance in this section, we therefore choose to concentrate on the factoid task of run `asked05b` since, while the performance of run `asked05c` was the best overall, it is a combination of outputs from several systems which makes it is less clear where errors originated and is therefore more difficult to analyse.

In Table 3 we give the percentage of errors (i.e. wrong and inexact answers as judged by NIST) for run `asked05b` on questions in the evaluation set that can be attributed to the retrieval, filter or a combination of retrieval and filter models. For this analysis we call an error anything that was marked wrong or inexact, of which there were 271 such errors.

Percentage of errors in each model combination				NOT
R	F	R&F	NK	ERR.
41.3%	28.0%	24.4%	5.6%	0.7%

Table 3. Percentage of errors of total 271 in *R*etrieval and *F*ilter models, *N*ot *K*nown errors, and *NOT* actually *ERR*ors for run `asked05b` on the TREC2005 factoid task.

It is clear that, even given this subjective evaluation of the errors, the retrieval model is mostly to blame. This is hardly surprising given the simplicity of our retrieval model. For example, we got almost all questions wrong that contained a *"did..."* construction such as *"When did X die?"* since the verb is almost always in a form different to that in the text where answers are likely to occur e.g. *"X died in 1974"*. Such questions accounted for almost 20% of the total set of factoid questions this year.

A large deficiency of model TWO used in the `asked05b` run is that numbers are not always assigned an equal probability by the filter model. Actually this applies equally to any ostensibly similar class of answers but the differences are most apparent for numbers. Approximately 19% of errors in the filter model could be attributed to this. The percentage is high partly because the number of questions this year which could be answered correctly with only a number was also very high—approximately 29% of questions, with 36% having a number somewhere in the correct answer.

Our system never output NIL as an answer. We preferred instead to output an answer whether or not support could be found for the answer. This year about 5% of questions required a NIL answer to be marked correct so we got them all wrong for all runs.

Despite the time difference between data in the AQUAINT corpus and the web data we were using very few errors were caused by this difference—only about 2% of errors.

There were also 2 answers in our output that were classified as wrong although we believe the document supports the answer which would actually make them either right or inexact rather than wrong. For question 100.1 *"Sammy Sosa": "Where was Sammy Sosa born?"* we gave the answer "SAN PEDRO DE MACORIS" in document NYT19980927.0104 from:

> ***SAN PEDRO de MACORIS****, Dominican Republic - As* ***Sammy Sosa*** *came to bat for the final time Sunday, the crowd of about 150 men in what may have been the only place* ***here, in his hometown*** *with a...*

For question 121.2 *"Rachel Carson": "Where was her home?"* we gave the answer "MAINE" in document NYT19991230.0073 from:

> *...1962.* ***Maine*** *biologist* ***Rachel Carson****..."*

A breakdown of the inexact answers showed that 9 errors were in location questions where a state was given but no town (or vice-versa); 6 errors were in time questions where only a year was given but a day and month was also required; and 4 errors were in names of people where a surname but no first name was given. From our system's point-of-view these were not errors since the q-and-a examples used in training (including those from previous TREC evaluations) also contained equally inexact answers but had been classified as correct. In future we will endeavour to remove potentially inexact training examples or replace them with more exact equivalent answers.

For the list question task it turned out to be somewhat naive to take the top-scoring answers from our factoid question answering system since many irrelevant and inappropriate answers were output as a result. Consequently it probably would have been better to output more answers rather than the 1 to 15 answers that were output for list questions; for example, run `asked05c` performed best and also had the largest average number of answers per question. A substantial cause of the poor list question performance was that there were far fewer list training q-and-a examples than those available for factoid training resulting in worse question-matching and therefore worse answer-matching performance. This matching was further muddied by the inclusion of factoid questions in the q-and-a set since the factoid question types are substantially different to list question types cf. use of singular vs. plurals in list questions. In future experiments on list questions we will restrict ourselves to using only list q-and-a as examples.

For the *other* question task, run `asked05c` (F-measure of 0.131) was quite similar to run `asked05a` (F-measure of 0.138), since it was made by taking the best answers from the `asked05a` run, and completing with nuggets from run `asked05b` up to a maximum of 18 nuggets. Run `asked05a,c` performed better than run `asked05b` (F-measure of 0.091), mostly because the web data that was used contained many garbage tokens that had not been cleaned correctly.

Projecting answers obtained from web data back on to the AQUAINT corpus documents turned out to be far from trivial. Indeed we lost around 20% of our correct, exact answers for runs `asked05b,c` because they were unsupported by the document we provided. Had the support been correct our best score would have been 26.5% for run `asked05c` on the factoid task. For the answers obtained only using the AQUAINT corpus the projection operation worked better and the loss was only around 13%.

Finally for run `asked05c` our system combination method was found to be surprisingly effective and robust despite being very simple. An absolute improvement in accuracy of 1.4% (or 7.0% relative) over our best individual run (`asked05b`) was obtained on the factoid task and 21.1% relative F-score improvement on the list task.

9 Conclusion

We have described our novel, data-driven and non-linguistic approach to question answering and presented the official results obtained in the TREC2005 evaluation. We have shown that our method, despite being very different to contemporary approaches achieves performance on the factoid task that is better than the majority of other systems. However, such performance is still substantially worse than the best participating systems.

We aim to extend our data-driven approach by including minimal linguistic transformations of the question such as verb-tense modification and term re-ordering such as performed by Aranea [10] and other systems. We will also demonstrate that our approach achieves similar performance on other languages when sufficient and suitable training data is available.

10 Online demonstration

A demonstration of the system using model ONE supporting questions in English, Japanese, Chinese, Russian and Swedish can be found online at `http://asked.jp/`

11 Acknowledgments

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

References

[1] 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.

[2] 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.

[3] 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.

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

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

[6] E. Hovy, U. Hermjakob, and L. C-Y. The Use of External Knowledge in Factoid QA. In *Proceedings of the TREC 2001 Conference*, 2001.

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

[8] T. Kikuchi, S. Furui, and C. Hori. Automatic speech summarization based on sentence extraction and compaction. In *Proceedings of ICASSP*, Hong Kong, China, 2003.

[9] J. Lin and D. Demner-Fushman. Automatically Evaluating Answers to Definition Questions. Technical Report LAMP-TR-119/CS-TR-4695/UMIACS-TR-2005-04, University of Maryland, 2005.

[10] J. Lin and B. Katz. Question Answering from the Web Using Knowledge Annotation and Knowledge Mining Techniques. In *Proceedings of Twelfth International Conference on Information and Knowledge Management (CIKM 2003)*, 2003.

[11] D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, 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. Whittaker, S. Furui, and D. Klakow. A Statistical Pattern Recognition Approach to Question Answering using Web Data. In *Proceedings of Cyberworlds*, 2005.