Monolingual Web-based Factoid Question Answering in Chinese, Swedish, English and Japanese

E.W.D. Whittaker **J. Hamonic** **D. Yang** **T. Klingberg** **S. Furui**
Dept. of Computer Science
Tokyo Institute of Technology
2-12-1, Ookayama, Meguro-ku
Tokyo 152-8552 Japan
{edw,yuuki,raymond,tor,furui}@furui.cs.titech.ac.jp

Abstract

In this paper we extend the application of our statistical pattern classification approach to question answering (QA) which has previously been applied successfully to English and Japanese to develop two prototype QA systems in Chinese and Swedish. We show what data is necessary to achieve this and also evaluate the performance of the two new systems using a translation of the TREC 2003 factoid QA task. While performance for Chinese and Swedish is found to be lower than that for the more developed English and Japanese systems we explain why this is the case and offer solutions for their improvement. All systems form the basis of our publicly accessible web-based multilingual QA system at `http://asked.jp`.

1 Introduction

Much of the research into automatic question answering (QA) has understandably concentrated on the English language with little regard to portability or efficacy in other languages. It is only relatively recently, with the introduction of the CLEF and NTCIR QA evaluations, that researchers have started to look at porting and evaluating the techniques that have been shown to work well for English to other languages.

One of the major drawbacks of porting an English language QA system or approach to other languages is often the lack of the corresponding NLP tools in the target language. For instance, parsers and named-entity (NE)-taggers, which are typical components in many QA systems, are certainly not available for all the world's languages. Trainable parsers and NE-taggers similarly require appropriate training data which, if not available, is costly and requires specialized knowledge to produce. Language-specific databases are also a common feature of many systems, some of which have taken many man-years to construct and verify. Such a component in many English-language systems, for example, is WordNet. While porting WordNet to other languages has been started in the Euro and Global WordNet projects they still only cover a relatively small number of languages.

In this paper we describe the application of our data-driven approach to QA which was developed right from the outset with the aim of portability and robustness in mind. This statistical pattern classification approach to QA is essentially language independent and trainable given appropriate language-specific training data. No assumptions about the language are made by the model except that some notion of words or space-separated tokens must exist or be introduced where it is absent. Our only other requirements to build a QA system in a new target language are: a large collection of text data in the target language that can be searched for answers (e.g. the web), and a list of example questions-and-answers (q-and-a) in the target language. Given these data sources the remaining components can be obtained automatically for each language.

So, in contrast to other contemporary approaches to QA our English language system does not use WordNet as in (Hovy et al., 2001; Moldovan et al., 2002), NE extraction, or any other linguistic information e.g. from semantic analysis (Hovy et al., 2001) or from question parsing (Hovy et al., 2001; Moldovan et al., 2002) and uses capitalised (where appropriate for the language) word tokens as the only features for modelling. For our Japanese system, although we currently use Chasen to segment Japanese character sequences into units that resemble words, we make no use of any morphological information as used for example in (Fuchigami et al., 2004). Moreover, it should be noted that our approach is at the same time very different to other purely web-based approaches such as askMSR (Brill et al., 2002) and Aranea (Lin et al., 2002). 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. These basic principles apply to each of our language-specific QA systems thus simplifying and accelerating development.

The approach to QA that we adopt has previously been described in (Whittaker et al., 2005a; Whittaker et al., 2005b; Whittaker et al., 2005c) where the details of the mathematical model and how it was trained for English and Japanese were given. Our approach has also been successfully evaluated in the text retrieval conference (TREC) 2005 QA track evaluation (Voorhees, 2003) where our group placed eleventh out of thirty participants (Whittaker et al., 2005a). Although the TREC QA task is substantially different to web-based QA the TREC evaluation confirmed that our approach works and also provides an objective assessment of its quality. Similarly, for our Japanese language system we have previously evaluated the performance of our approach on the NTCIR-3 QAC-1 task (Whittaker et al., 2005c). Although our Japanese experiments were applied retrospectively, the results would have placed us in the mid-range of participating systems in that year's evaluation. In this paper we present additional experiments on Chinese and Swedish and explain how our statistical pattern classification approach to QA was successfully applied to these two new languages. Using our approach and given appropriate training data it is found that a reasonably proficient developer can build a QA system in a new language in around 10 hours. Evaluation of the Chinese and Swedish systems is performed using a translation of the first 200 factoid questions from the TREC 2003 evaluation which we have also made available online. We compare these results both qualitatively and quantitatively against results obtained previously for English and Japanese. The systems, built using this method, form the basis of our multi-language web demo which is publicly available at `http://asked.jp`.

An outline of the remainder of this paper is as follows: we briefly describe our statistical pattern classification approach to QA in Section 2 repeating the important elements of our approach as necessary to understand the remainder of the paper. In Section 3 we describe the basic building blocks of our QA system and how they can typically be trained. We also give a breakdown of the data used to train each language specific QA system. In Section 4 we present the results of experiments on Chinese, Swedish, English and Japanese and in Section 5 we compare and analyse these results. We wrap up with a conclusion and further work in Sections 6 and 7.

2 Statistical pattern classification approach to QA

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 NTCIR and 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 *who, when, where, which,* 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:

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

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 (see (Whittaker et al., 2005a) for more details):

$$\arg\max_A \underbrace{P(A \mid X)}_{\substack{retrieval\\model}} \cdot \underbrace{P(W \mid A)}_{\substack{filter\\model}}. \qquad (3)$$

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. 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 modelling 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 *stoplist* of 50-100 high-frequency words. For more details on the exact form of the retrieval model please refer to (Whittaker et al., 2005a).

2.2 Filter model

A set of $|V_{\mathcal{W}}|$ single-word features is extracted based on frequency of occurrence in question data. Some examples include: *HOW*, *MANY*, *WHEN*, *WHO*, *UNTIL* etc. 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 MANY* and *UNTIL WHEN*.

Modelling 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 modelling we say that W is conditionally independent of A given c_e as follows:

$$P(W \mid A) = \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot P(c_e \mid A). \qquad (4)$$

Given a set E of example q-and-a t_j for $j = 1 \ldots |E|$ where $t_j = (q^j, a^j) = (q_1^j, \ldots, q_{l_{Qj}}^j, a_1^j, \ldots, a_{l_{Aj}}^j)$ we define a mapping function $f : E \to C_E$ by $f(t_j) = e$. Each class $c_e = (w_1^e, \ldots, w_{l_{We}}^e, a_1^e, \ldots, a_{l_{Ae}}^e)$ is then obtained by $c_e = \bigcup_{j:f(t_j)=e} \mathcal{W}(q^j) \bigcup_{i=1}^{l_{Aj}} a_i^j$. In all the experiments in this paper no clustering of the q-and-a is actually performed so each q-and-a example forms its own unique class i.e. each c_e corresponds to a single t_j and vice-versa.

Assuming conditional independence of the answer words in class c_e given A and making the modelling 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:

$$\begin{aligned} P(W \mid A) &= \sum_{e=1}^{|C_E|} P(W \mid c_e) \cdot \prod_{j=1}^{l_{Ae}} P(a_j^e \mid a_j) \\ &= \sum_{e=1}^{|C_E|} P(W \mid c_e) \prod_{j=1}^{l_{Ae}} \sum_{a=1}^{|C_A|} P(a_j^e \mid c_a) P(c_a \mid a_j), \end{aligned} \qquad (5)$$

where c_a is a concrete class in the set of $|C_A|$ answer classes C_A, and assuming a_j^e is conditionally independent of c_a given a_j. The system using the formulation of filter model given by Equation (5) is referred to as model ONE. The model given by Equation (4) is referred to as model TWO, however, we are only concerned with model ONE in this paper.

Answer typing, such as it exists in our model ONE system, is performed by the filter model and is effected by matching the input query against our set of example questions C_E. Each one of these example questions c_e has an answer associated with it which in turn is expanded via the C_A classes into a set of possible answers for each question. At each step this matching process is entirely probabilistic as given by Equation (5). To take a simple example, if we are presented with the input query *"Who was the U.S. President who first used Camp David?"* the first step effectively matches the words (and bigrams, trigrams) in that query against the words (and bigrams, trigrams) in our example questions with a probability of its match assigned to all example questions. Suppose, for example, that the top-scoring example question in our set is *"Who was the U.S. President who resigned as a result of the Watergate affair?"* since the first six words in each question match each other and will likely result in a high probability being assigned. The corresponding answer to this example question is *"Richard Nixon"*. So the next step expands *"Richard"* using C_A and results in a high score being assigned to a class of words which tend to share the feature of being male first names, one of which is *"Franklin"*. Expanding the second word in the answer *"Nixon"* using C_A will possibly result in a high score being assigned to a class of words that share the feature of being the surnames of U.S. presidents, one of which is *"Roosevelt"*. In this way, we end up with a high score assigned by the filter model to *"Franklin Roosevelt"* but also to *"Abraham Lincoln"*, *"Bill Clinton"*, etc. In combination with the retrieval model, we hope that the documents obtained for this query assign a higher retrieval model score to *"Franklin Roosevelt"* over the names of other U.S. presidents and thus output it in first place with the highest overall probability. While this approach works well for names, dates and short place names it does fall down, for example, on names of books, plays and films where there is typically less of a clear correspondence between the words in any given position of two answers. This situation could be avoided by using multi-word answer strings and not making the position-dependence modelling assumption that was made to arrive at Equation (5) but this has its own drawbacks.

The above description of the operation of the filter model highlights the need for homogeneous classes of C_A of sufficiently wide coverage. In the next section we describe a way in which this can be achieved in an efficient, data-driven manner for essentially any language.

2.3 Obtaining C_A

As we saw in the previous section the C_A are the closest thing we have to named entities since they define classes of words that share some similarity with each other. However, in some sense they are more flexible than named entities since any word can actually belong to any class but with a certain probability. In this way we don't rule out with a zero probability the possibility of a word belonging to some class, just that a word is more likely to belong to some classes than others. In addition, the entities are not actually named i.e. we do not impose our own label on the classes so we do not explicitly have a class of first names, or a class of days of the week. Although we hope that we will end up with classes containing such similar words we do not make it explicit and we do not label what each class of words is supposed to represent.

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 set of potential answer words V_{C_A} that are clustered, should ideally cover all possible words that might ever be answers to questions. We therefore take the most frequent $|V_{C_A}|$ words from a language-specific corpus T comprising $|T|$ word tokens as our set of potential answers. The "seeds" for the clusters are chosen to be the most frequent $|C_A|$ words in T. The algorithm then uses the co-occurrence probabilities of words in T to group together words with similar co-occurrence statistics. For each word a in V_{C_A} 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:

$$\begin{aligned}\vec{a} = [&P(a_1 \mid a, 1), \ldots, P(a_R \mid a, 1), P(a_1 \mid a, -1), \ldots, \\ &P(a_R \mid a, -1), \ldots\ldots, P(a_1 \mid a, D), \ldots, P(a_R \mid a, D), \\ &P(a_1 \mid a, -D), \ldots, P(a_R \mid a, -D)]. \quad (6)\end{aligned}$$

Rather than storing $2DV_{C_A}^2$ elements we can compute most terms efficiently and on-the-fly using

Language	T	$\|T\|$	$\|V_{C_A}\|$	$\|C_E\|$	$\|C_A\|$
Chinese	TREC Mandarin (Rogers, 2000)	68M	33k	7k	1000
Swedish	PAROLE (University, 1997)	19M	367k	5k	1000
English	AQUAINT (Voorhees, 2002)	300M	215k	290k	500
Japanese	MAINICHI (Fukumoto et al., 2002)	150M	300k	270k	5000

Table 1: Description of each of the four monolingual QA systems.

the Katz back-off method (Katz, 1987) 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: $\mathcal{D}(\vec{a_i}, \vec{a_j}) = |\vec{a_i} - \vec{a_j}|$. Merging two vectors then involves a straightforward update of the co-occurrence counts for a cluster and recomputing 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 $\mathcal{D}(\vec{a_i}, \vec{c_j})$
   move $\vec{a_i}$ to $\vec{c}_j^{best}$
   update $\vec{c}_j^{best}$
```

Although the algorithm is currently applied offline before system deployment it could also be easily adapted to handle multi-word sequences and could also be applied at run-time thus reducing the effects of out-of-vocabulary answers.

3 System components

There are four basic ingredients to building a QA system using our approach: (1) a collection of example q-and-a (C_E) pairs used for answer-typing (answers need not necessarily be correct but must be of the correct answer type); (2) a classification of words (or word-like units cf. Japanese/Chinese) into classes of similar words (C_A) e.g. a class of country names, of given names, of numbers etc.; (3) a list of question words (*qlist*) such as *"WHO"*, *"WHERE"*, *"WHEN"* etc.; and (4) a stop list of words that should be ignored by the retrieval model (*stoplist*) as described in Section 2.1.

The q-and-a (C_E) for different languages can often be found on the web or in commercial quiz software. To date, we have not examined how many or what distribution of types of example q-and-a are important for good performance. However, intuitively, one expects that the richer and more varied the questions the better the performance will be. For the time being we aim to include as many examples as possible and hope this covers the kinds of questions that are asked in reality (or at least in the evaluations). In principle, in a working system it should also be possible to feed a user's question together with the user's response back into the system to improve performance in an unsupervised manner, so that performance would gradually improve over time.

To obtain the classes of answers C_A for each language Algorithm 2.1 is applied. The qlist is generated by taking the most frequently occurring terms from the question portion of the example q-and-a. The stoplist is formed from the 50-100 most frequently occurring words in T.

Google is used to retrieve documents for all questions, except Japanese, where an index of the NTCIR 2001 web snapshot is used instead. The question is passed as-is to Google after the removal of stop words. The top U documents are downloaded in their entirety, HTML markup removed, the text cleaned and upper-cased (where appropriate). For consistency, all data in our system is encoded using UTF-8.

The data source and relevant system details for each language-specific QA system are given in Table 1. For the Japanese system Chasen[1] is used to segment character sequences into word-like units. For Chinese each sentence is mapped to a sequence of space-separated characters.

4 Experimental work

A substantial amount of time has gone into investigating combinations and ranges of parameters which gave good performance on English development questions. These same parameters were largely used as-is for the Japanese system with only minor optimisations performed on de-

[1] `http://chasen.naist.jp/hiki/ChaSen`

Top Answers	Chinese "TREC 2003"	Swedish "TREC 2003"	English		Japanese QAC-1
			TREC 2003	TREC 2005	
1	14 (7%)	23 (11.5%)	99 (24.0%)	64 (17.7%)	37 (18.5%)
5	24 (12.0%)	34 (17.0%)	175 (42.4%)	121 (33.4%)	61 (30.5%)
10	33 (16.5%)	46 (23.0%)	196 (47.5%)	151 (41.7%)	69 (34.5%)
20	39 (19.5%)	50 (25.0%)	216 (52.3%)	167 (46.1%)	81 (40.5%)
Total Questions	200	200	413	362	200

Table 2: Performance of each language-specific system in the top 1, 5, 10 and 20 answers output.

velopment questions disjoint from any evaluation set. For the two new QA systems in Chinese and Swedish we used exactly the same parameters that had been found to work well for the English system and performed no optimisation, mainly due to a lack of questions that we could hold-out for development. This is clearly sub-optimal but will be addressed in future work.

It has been found for English and Japanese that system performance consistently increases the more documents that are used to search for an answer. Experiments with English used $U = 500$ documents and for Japanese $U = 5000$ documents was used. Due to time constraints, however, for the Swedish experiments we used a maximum of $U = 100$ documents. For the Chinese experiments we were forced to use a maximum of only $U = 20$ documents. This was because the segmentation of Chinese text into individual characters resulted in a huge number of character combinations being generated in the retrieval model and due to memory limitations the only way round this problem was to limit the number of documents used. In addition, for an answer to be output the number of times it had to occur in the data was set to 5 for Swedish and 2 for Chinese. These values were based on the number of documents we were using and our prior experience with English and Japanese.

4.1 Test data

For evaluating the English and Japanese systems there were obvious candidates for evaluation sets well-known to the QA community since the TREC and NTCIR QA evaluations have been running now for several years. For the English experiments we performed two sets of experiments using the TREC 2003 and TREC 2005 factoid QA questions. For the Japanese experiments we used the formal run of the NTCIR-3 QAC-1 task.

For Chinese and Swedish there were no such standard test sets available. We therefore decided to translate the first 200 questions in the TREC 2003 QA factoid task into Chinese and Swedish. For Swedish this was relatively straightforward. For Chinese, we didn't know the Chinese translation of the names, places etc. for about 40% of the original TREC questions. Such questions were therefore "translated" into a roughly equivalent question about something similar in China. However, the final set of Chinese questions still contained 17% of questions that had a U.S. background. Correct answers were found by a human to 91% and 93% of questions for Chinese and Swedish, respectively. We hope that using the TREC questions should at least give the reader some idea of the kind of the questions that were asked so they can appreciate the difficulty of finding the answers in Chinese and Swedish web data. To aid comprehension of the task we are tackling we have made the questions and the answers we used for evaluation freely available[2] for anyone else interested in running similar experiments.

4.2 Results

The TREC, NTCIR and CLEF QA evaluations have all converged on the use of exact and supported answers as a metric of QA system performance. In web-based QA we are also concerned with document support but primarily with answer correctness so for now we only use answer correctness as our evaluation metric. However, we also adopt the notion of inexact answers to mark as incorrect, answers that have extraneous or missing parts such as units of measurement or in the case of, say Chinese, an extra or missing character.

In Table 2 we present the number and percentage of answers that were marked correct in the top 1, 5, 10 and 20 answers output by each of the four monolingual systems.

[2] `http://asked.jp/About.html`

5 Discussion

Probably the first observation one makes looking at Table 2 is that the performance of the Chinese and Swedish systems is significantly lower than the English and Japanese systems. On the face of it this is not so surprising since far more documents and many more hours went into developing the English and Japanese systems for which the system parameters were also optimised on held-out question data prior to evaluation.

One significant issue common to our evaluation of the Chinese and Swedish systems is the use of a translation of the TREC 2003 questions. Although it was hoped this would aid comprehension of the task we were attempting it had the predictable problem that not enough, or any (in some cases), relevant Chinese and Swedish web data could be found to answer some questions. Particularly problematic were U.S. centric questions about baseball and U.S. geography of which there were around 5% and 10%, respectively, in the Swedish test set. However, aside from the choice of test data there are several other possible reasons for the drop in performance and we'll examine them separately for Chinese and Swedish in the following sections.

5.1 Chinese

One of the most important issues in building a Chinese QA system was deciding on the segmentation for text in which there are no spaces between words. In contrast to our Japanese system where we used a freely available segmentation tool we chose to segment Chinese text into single characters and to not even try to recover a notion of words from the text. Clearly this is far from satisfactory but the redundancy aspect of our approach does compensate for this to some extent by favouring frequent character sequences which are more likely to represent words. The second major problem we had was downloading sufficient and relevant data for each question. The largest culprit here was our restriction on using only 20 documents, the reasons for which were explained in Section 4. However, the quality of the documents themselves was also a problem: only 47.5% of questions had data in which the correct answer was observed at least once in the data and this was reduced to 43.5% when an answer had to occur 2 or more times. This places a much lower value on the performance upper bound (i.e. 47.5%) compared to English where it is typically 80-90% and consequently puts the results we obtained in a more positive light. Another interesting observation was that there were very few inexact answers (10% in the top 20) which shows that the modelling assumption made in Equation (5) was not detrimental for Chinese even when segmenting by characters.

5.2 Swedish

The general structure of Swedish and its character set are very similar to English so there were very few modifications that needed to be made to port the English system to Swedish. In some senses Swedish (and indeed many other European languages) are much easier than English for a purely statistical keyword-based system since they do not have the same phenomenon that English has where many[3] questions use *does/did* as in "*Who/when/where does/did ...?*" etc. questions. Swedish does, however, suffer from some of the same problems as English such as the need to re-order words in a question to produce a more statement-like formulation. A more significant problem is that Swedish is a language spoken by only about 9 million people and correspondingly there were far fewer training questions and answers freely available on the web and much less web-data available for finding answers. Although our maximum number of documents was set to 100, we only downloaded 48 documents per question on average (s.d. of 30). In a data-driven approach that relies to a large extent on data redundancy and the philosophy that "the more data the better" to achieve good performance this is always likely to be problematic. Indeed, of the documents that were downloaded for each question 67.5% contained a correct answer. This was reduced to 50% for correct answers occuring the threshold number of 5 times or more. Clearly a more linguistics-based system suffers less from this lack of redundancy but as pointed out earlier the problem is simply transferred to the development stage instead, where the initial data or linguistic knowledge needed to train a system might well be lacking.

6 Conclusion

In this paper we have shown that usable performance from two prototype QA systems developed

[3]For example, 28% of TREC 2003 factoid questions.

for Chinese and Swedish could be obtained in a matter of hours using our data-driven approach to question answering. While the performance of such systems was found to be below that of more developed systems that used the same approach we feel that this demonstrates the effectiveness of our language independent approach for web-based QA. One of the interesting observations from our experiments is the inherent difficulty of finding answers to questions when there is very little data available in the target language in which to search for answers. While one could argue that someone in Sweden or China may not be interested in finding out the winning team of the 1996 World Series this clearly misses the point and instead makes a compelling argument for cross-language QA (CLQA) whose profile has been steadily increasing due to the recent CLEF and NTCIR evaluations. While most attempts at CLQA have so far concentrated on translating the query into English and performing monolingual English QA on the translated query, we feel that an approach that also combines a simple system trained in the manner described in this paper would almost certainly benefit overall performance.

7 Further work

In the future we aim to improve the Chinese and Swedish systems by increasing the amount and quality of data used for training and also the data used for searching for answers as well as solving the explosion in memory usage that occurred with large amounts of Chinese text. We also intend to optimise the system parameters for each system using the questions and answers used for training with a rotating form of cross-validation so as to maximise use of the little data we have available. Given the rapid development time and reasonable performance of the approach outlined in this paper combined with the fact that specialized linguistic knowledge is unnecessary we also plan the development of yet more monolingual systems in other languages and hope to participate in the upcoming CLEF QA track.

8 Acknowledgments

This research was supported by JSPS and the Japanese government 21st century COE programme. The authors also with to thank Dietrich Klakow for his contributions to the model development.

References

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

M. Fuchigami, H. Ohnuma, and A. Ikeno. 2004. Oki QA System for QAC-2. In *Proceedings of NTCIR-4 Workshop*.

J. Fukumoto, T. Kato, and F. Masui. 2002. Question Answering Challenge (QAC-1) An Evaluation of Question Answering Task at NTCIR Workshop 3. In *Proceedings of NTCIR-3 Workshop*.

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

S. M. Katz. 1987. Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustic, Speech, and Signal Processing*, 35(3):400–401.

J. Lin, A. Fernandes, B. Katz, G. Marton, and S. Tellex. 2002. Extracting Answers from the Web Using Knowledge Annotation and Knowledge Mining Techniques. In *Proceedings of the Eleventh Text REtrieval Conference (TREC2002)*, Gaithersburg, Maryland, November.

D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, and O. Bolohan. 2002. LCC Tools for Question Answering. In *Proceedings of the TREC 2002 Conference*.

Willie Rogers. 2000. TREC Mandarin. Linguistic Data Consortium.

Sprakbanken Gothenburg University. 1997. PAROLE Swedish Language Corpus. `http://spraakbanken.gu.se/lbeng.html`.

E. Voorhees. 2002. Overview of the TREC 2002 Question Answering Track. In *Proceedings of the TREC 2002 Conference*.

E. Voorhees. 2003. Overview of the TREC 2003 Question Answering Track. In *Proceedings of the TREC 2003 Conference*.

E.W.D. Whittaker, P. Chatain, S. Furui, and D. Klakow. 2005a. TREC2005 Question Answering Experiments at Tokyo Institute of Technology. In *Proceedings of the 14th Text Retrieval Conference*.

E.W.D. Whittaker, S. Furui, and D. Klakow. 2005b. A Statistical Pattern Recognition Approach to Question Answering using Web Data. In *Proceedings of Cyberworlds*.

E.W.D. Whittaker, J. Hamonic, and S. Furui. 2005c. A Unifi ed Approach to Japanese and English Question Answering. In *Proceedings of NTCIR-5*.