Extracting Key Semantic Terms from Chinese Speech Query for Web Searches

Gang WANG
National University of Singapore
wanggang_sh@hotmail.com

Tat-Seng CHUA
National University of Singapore
chuats@comp.nus.edu.sg

Yong-Cheng WANG
Shanghai Jiao Tong University, China, 200030
ycwang@mail.sjtu.edu.cn

Abstract

This paper discusses the challenges and proposes a solution to performing information retrieval on the Web using Chinese natural language speech query. The main contribution of this research is in devising a divide-and-conquer strategy to alleviate the speech recognition errors. It uses the query model to facilitate the extraction of main core semantic string (CSS) from the Chinese natural language speech query. It then breaks the CSS into basic components corresponding to phrases, and uses a multi-tier strategy to map the basic components to known phrases in order to further eliminate the errors. The resulting system has been found to be effective.

1 Introduction

We are entering an information era, where information has become one of the major resources in our daily activities. With its wide spread adoption, Internet has become the largest information wealth for all to share. Currently, most (Chinese) search engines can only support term-based information retrieval, where the users are required to enter the queries directly through keyboards in front of the computer. However, there is a large segment of population in China and the rest of the world who are illiterate and do not have the skills to use the computer. They are thus unable to take advantage of the vast amount of freely available information. Since almost every person can speak and understand spoken language, the research on "(Chinese) natural language speech query retrieval" would enable average persons to access information using the current search engines without the need to learn special computer skills or training. They can simply access the search engine using common devices that they are familiar with such as the telephone, PDA and so on.

In order to implement a speech-based information retrieval system, one of the most important challenges is how to obtain the correct query terms from the spoken natural language query that convey the main semantics of the query. This requires the integration of natural language query processing and speech recognition research.

Natural language query processing has been an active area of research for many years and many techniques have been developed (Jacobs and Rau1993; Kupie, 1993; Strzalkowski, 1999; Yu et al, 1999). Most of these techniques, however, focus only on written language, with few devoted to the study of spoken language query processing.

Speech recognition involves the conversion of acoustic speech signals to a stream of text. Because of the complexity of human vocal tract, the speech signals being observed are different, even for multiple utterances of the same sequence of words by the same person (Lee et al 1996). Furthermore, the speech signals can be influenced by the differences across different speakers, dialects, transmission distortions, and speaking environments. These have contributed to the noise and variability of speech signals. As one of the main sources of errors in Chinese speech recognition come from substitution (Wang 2002; Zhou 1997), in which a wrong but similar sounding term is used in place of the correct term, confusion matrix has been used to record confused sound pairs in an attempt to eliminate this error. Confusion matrix has been employed effectively in spoken document retrieval (Singhal et al, 1999 and Srinivasan et al 2000) and to minimize speech recognition errors (Shen et al, 1998). However, when such method is used directly to correct speech recognition errors, it tends to bring in too many irrelevant terms (Ng 2000).

Because important terms in a long document are often repeated several times, there is a good chance that such terms will be correctly recognized at least once by a speech recognition engine with a reasonable level of word recognition rate. Many spoken document retrieval (SDR) systems took advantage of this fact in reducing the speech recognition and matching errors (Meng et al 2001; Wang et al 2001; Chen et al 2001). In contrast to SDR, very little work has been done on Chinese spoken query processing (SQP), which is the use of spoken queries to retrieval textual documents. Moreover, spoken queries in SQP tend to be very short with few repeated terms.

In this paper, we aim to integrate the spoken language and natural language research to process spoken queries with speech recognition errors. The main contribution of this research is in devising a divide-and-conquer strategy to alleviate the speech recognition errors. It first employs the Chinese query model to isolate the Core Semantic String (CSS) that conveys the semantics of the spoken query. It then breaks the CSS into basic components corresponding to phrases, and uses a multi-tier strategy to map the basic components to known phrases in a dictionary in order to further eliminate the errors.

In the rest of this paper, an overview of the proposed approach is introduced in Section 2. Section 3 describes the query model, while Section 4 outlines the use of multi-tier approach to eliminate errors in CSS. Section 5 discusses the experimental setup and results. Finally, Section 6 contains our concluding remarks.

2 Overview of the proposed approach

There are many challenges in supporting surfing of Web by speech queries. One of the main challenges is that the current speech recognition technology is not very good, especially for average users that do not have any speech trainings. For such unlimited user group, the speech recognition engine could achieve an accuracy of less than 50%. Because of this, the key phrases we derived from the speech query could be in error or missing the main semantic of the query altogether. This would affect the effectiveness of the resulting system tremendously.

Given the speech-to-text output with errors, the key issue is on how to analyze the query in order to grasp the Core Semantic String (CSS) as accurately as possible. CSS is defined as the key term sequence in the query that conveys the main semantics of the query. For example, given the query: "请问有什么关于美国将中国的人权状况与其是否给予中共最惠国待遇分离的消息" (Please tell me the information on how the U.S. separates the most-favored-nation status from human rights issue in china). The CSS in the query is underlined. We can segment the CSS into several basic components that correspond to key concepts such as: 美国 (U.S.), 中国 (China), 人权状况 (human rights issue), 最惠国待遇 (the most-favored-nation status) and 分离 (separate).

Because of the difficulty in handling speech recognition errors involving multiple segments of CSSs, we limit our research to queries that contain only one CSS string. However, we allow a CSS to include multiple basic components as depicted in the above example. This is reasonable as most queries posed by the users on the Web tend to be short with only a few characters (Pu 2000).

Thus the accurate extraction of CSS and its separation into basic components is essential to alleviate the speech recognition errors. First of all, isolating CSS from the rest of speech enables us to ignore errors in other parts of speech, such as the greetings and polite remarks, which have no effects on the outcome of the query. Second, by separating the CSS into basic components, we can limit the propagation of errors, and employ the set of known phrases in the domain to help correct the errors in these components separately.

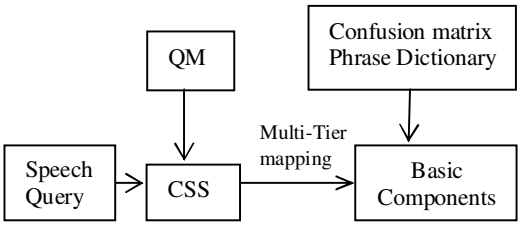


Figure 1: Overview of the proposed approach

To achieve this, we process the query in three main stages as illustrated in Figure 1. First, given the user's oral query, the system uses a speech recognition engine to convert the speech to text. Second, we analyze the query using a query model (QM) to extract CSS from the query with minimum errors. QM defines the structures and some of the standard phrases used in typical queries. Third, we divide the CSS into basic components, and employ a multi-tier approach to match the ba-

sic components to the nearest known phrases in order to correct the speech recognition errors. The aim here is to improve recall without excessive lost in precision. The resulting key components are then used as query to standard search engine.

The following sections describe the details of our approach.

3 Query Model (QM)

Query model (QM) is used to analyze the query and extract the core semantic string (CSS) that contains the main semantic of the query. There are two main components for a query model. The first is query component dictionary, which is a set of phrases that has certain semantic functions, such as the polite remarks, prepositions, time etc. The other component is the query structure, which defines a sequence of acceptable semantically tagged tokens, such as "Begin, Core Semantic String, Question Phrase, and End". Each query structure also includes its occurrence probability within the query corpus. Table 2 gives some examples of query structures.

3.1 Query Model Generation

In order to come up with a set of generalized query structures, we use a query log of typical queries posed by users. The query log consists of 557 queries, collected from twenty-eight human subjects at the Shanghai Jiao Tong University (Ying 2002). Each subject is asked to pose 20 separate queries to retrieve general information from the Web.

After analyzing the queries, we derive a query model comprising 51 query structures and a set of query components. For each query structure, we compute its probability of occurrence, which is used to determine the more likely structure containing CSS in case there are multiple CSSs found. As part of the analysis of the query log, we classify the query components into ten classes, as listed in Table 1. These ten classes are called semantic tags. They can be further divided into two main categories: the closed class and open class. Closed classes are those that have relatively fixed word lists. These include question phrases, quantifiers, polite remarks, prepositions, time and commonly used verb and subject-verb phrases. We collect all the phrases belonging to closed classes from the query log and store them in the query component dictionary. The open class is the CSS, which we do not know in advance. CSS typically includes person's names, events and country's names etc.

Table 1: Definition and Examples of Semantic tags

Sem Tag	Name of tag	Example
1.	Verb-Object Phrase	给 (give) 我 (me)
2.	Question Phrase	有什么(is there)
3.	Question Field	新闻 (news), 报道 (report)
4.	Quantifier	一些 (some)
5.	Verb Phrase	找出 (find)，收集 (collect)
6.	Polite Remark	请帮我 (please help me)
7.	Preposition	关于 (about), 有关 (about)
8.	Subject-Verb phrase	我 (I) 要 (want)
9.	Core Semantic String	9.11 事件 (9.11 event)
10.	Time	今天 (today)

Table 2: Examples of Query Structure

1	Q1: 0, 2, 7, 9, 3, 0: 0.0025, 有什么 关于 9.11 事件的 新闻 2 7 9 3 Is there any information on September 11?
2	Q2: 0, 1, 7, 9, 3, 0 :0.01 给我 有关 本拉登的 报道 1 7 9 3 Give me some information about Ben laden.

Given the set of sample queries, a heuristic rule-based approach is used to analyze the queries, and break them into basic components with assigned semantic tags by matching the words listed in Table 1. Any sequences of words or phrases not found in the closed class are tagged as CSS (with Semantic Tag 9). We can thus derive the query structures of the form given in Table 2.

3.2 Modeling of Query Structure as FSA

Due to speech recognition errors, we do not expect the query components and hence the query structure to be recognized correctly. Instead, we parse the query structure in order to isolate and extract CSS. To facilitate this, we employ the Finite State Automata (FSA) to model the query structure. FSA models the expected sequences of tokens in typical queries and annotate the semantic tags, including CSS. A FSA is defined for each of the 51 query structures. An example of FSA is given in Figure 2.

Because CSS is an open set, we do not know its content in advance. Instead, we use the following

two rules to determine the candidates for CSS: (a) it is an unknown string not present in the Query Component Dictionary; and (b) its length is not less than two, as the average length of concepts in Chinese is greater than one (Wang 1992).

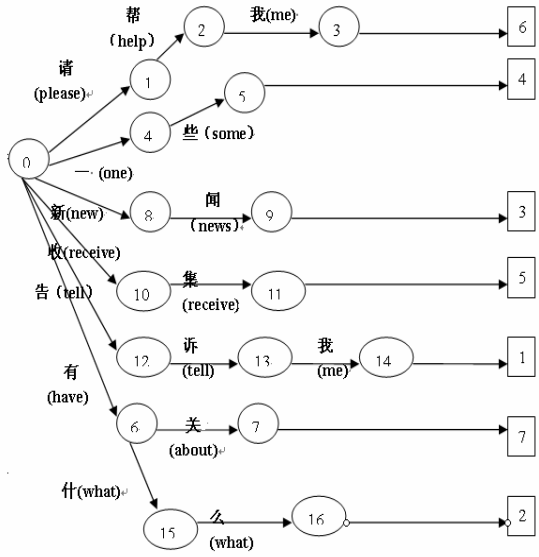
At each stage of parsing the query using FSA (Hobbs et al 1997), we need to make decision on which state to proceed and how to handle unexpected tokens in the query. Thus at each stage, FSA needs to perform three functions:

a) Goto function: It maps a pair consisting of a state and an input symbol into a new state or the fail state. We use G(N,X) =N' to define the goto function from State N to State N', given the occurrence of token X.
b) Fail function: It is consulted whenever the goto function reports a failure when encountering an unexpected token. We use f(N) =N' to represent the fail function.
c) Output function: In the FSA, certain states are designated as output states, which indicate that a sequence of tokens has been found and are tagged with the appropriate semantic tag.

To construct a goto function, we begin with a graph consisting of one vertex which represents State 0.We then enter each token X into the graph by adding a directed path to the graph that begins at the start state. New vertices and edges are added to the graph so that there will be, starting at the start state, a path in the graph that spells out the token X. The token X is added to the output function of the state at which the path terminates.

For example, suppose that our Query Component Dictionary consists of seven phrases as follows: "请帮我 (please help me); 一些 (some); 有关 (about); 新闻 (news); 收集 (collect); 告诉我 (tell me); 有什么 (what do you have)". Adding these tokens into the graph will result in a FSA as shown in Figure 2. The path from State 0 to State 3 spells out the phrase "请帮我 (Please help me)", and on completion of this path, we associate its output with semantic tag 6. Similarly, the output of "一些 (some)" is associated with State 5, and semantic tag 4, and so on.

We now use an example to illustrate the process of parsing the query. Suppose the user issues a speech query: "请帮我收集一些有关本拉登的新闻" (please help me to collect some information about Bin Laden). However, the result of speech recognition with errors is: "请 (please) 帮 (help) 我 (me) 收 (receive) 寄 (send) 一些 (some) 有关 (about) 半 (half) 拉 (pull) 灯 (light) 的 (of) 新闻 (news)". Note that there are 4 mis-recognized characters which are underlined.



Note : ☐ indicates the semantic tag.

Figure 2: FSA for part of Query Component Dictionary

The FSA begins with State 0. When the system encounters the sequence of characters 请 (please) 帮 (help)我 (me), the state changes from 0 to 1, 2 and eventually to 3. At State 3, the system recognizes a polite remark phrase and output a token with semantic tag 6.

Next, the system meets the character 收 (receive), it will transit to State 10, because of g(0, 收)=10. When the system sees the next character 寄 (send), which does not have a corresponding transition rule, the goto function reports a failure. Because the length of the string is 2 and the string is not in the Query Component Dictionary, the semantic tag 9 is assigned to token"收寄" according to the definition of CSS.

By repeating the above process, we obtain the following result:

请帮我	收寄	一些	有关	半拉灯	新闻
6	9	4	7	9	3

Here the semantic tags are as defined in Table 1. It is noted that because of speech recognition errors, the system detected two CSSs, and both of them contain speech recognition errors.

3.3 CSS Extraction by Query Model

Given that we may find multiple CSSs, the next stage is to analyze the CSSs found along with their surrounding context in order to determine the most probable CSS. The approach is based on the premise that choosing the best sense for an input vector amounts to choosing the most probable sense given that vector. The input vector i has three components: left context (L_i), the CSS itself (CSS_i), and right context (R_i). The probability of such a structure occurring in the Query Model is as follows:

$$s_i = \sum_{j=0}^{n} (C_{ij} * p_j) \quad (1)$$

where C_{ij} is set to 1 if the input vector i (L_i, R_i) matches the two corresponding left and right CSS context of the query structure j, and 0 otherwise. p_j is the possibility of occurrence of the j^{th} query structure, and n is the total number of the structures in the Query Model. Note that Equation (1) gives a detected CSS higher weight if it matches to more query structures with higher occurrence probabilities. We simply select the best CSS_i such that $\arg\max_i (s_i)$ according to Eqn(1).

For illustration, let's consider the above example with 2 detected CSSs. The two CSS vectors are: [6, 9, 4] and [7, 9, 3]. From the Query Model, we know that the probability of occurrence, p_j, of structure [6, 9, 4] is 0, and that of structure [7, 9, 3] is 0.03, with the latter matches to only one structure. Hence the s_i values for them are 0 and 0.03 respectively. Thus the most probable core semantic structure is [7, 9, 3] and the CSS "半 (half) 拉 (pull) 灯 (light)" is extracted.

4 Query Terms Generation

Because of speech recognition error, the CSS obtained is likely to contain error, or in the worse case, missing the main semantics of the query altogether. We now discuss how we alleviate the errors in CSS for the former case. We will first break the CSS into one or more basic semantic parts, and then apply the multi-tier method to map the query components to known phrases.

4.1 Breaking CSS into Basic Components

In many cases, the CSS obtained may be made up of several semantic components equivalent to base noun phrases. Here we employ a technique based on Chinese cut marks (Wang 1992) to perform the segmentation. The Chinese cut marks are tokens that can separate a Chinese sentence into several semantic parts. Zhou (1997) used such technique to detect new Chinese words, and reported good results with precision and recall of 92% and 70% respectively. By separating the CSS into basic key components, we can limit the propagation of errors.

4.2 Multi-tier query term mapping

In order to further eliminate the speech recognition errors, we propose a multi-tier approach to map the basic components in CSS into known phrases by using a combination of matching techniques. To do this, we need to build up a phrase dictionary containing typical concepts used in general and specific domains. Most basic CSS components should be mapped to one of these phrases. Thus even if a basic component contains errors, as long as we can find a sufficiently similar phrase in the phrase dictionary, we can use this in place of the erroneous CSS component, thus eliminating the errors.

We collected a phrase dictionary containing about 32,842 phrases, covering mostly base noun phrase and named entity. The phrases are derived from two sources. We first derived a set of common phrases from the digital dictionary and the logs in the search engine used at the Shanghai Jiao Tong University. We also derived a set of domain specific phrases by extracting the base noun phrases and named entities from the on-line news articles obtained during the period. This approach is reasonable as in practice we can use recent web or news articles to extract concepts to update the phrase dictionary.

Given the phrase dictionary, the next problem then is to map the basic CSS components to the nearest phrases in the dictionary. As the basic components may contain errors, we cannot match them exactly just at the character level. We thus propose to match each basic component with the known phrases in the dictionary at three levels: (a) character level; (b) syllable string level; and (c) confusion syllable string level. The purpose of matching at levels b and c is to overcome the homophone problem in CSS. For example, "拉登 (Laden)" is wrongly recognized as "拉灯 (pull lamp)" by the speech recognition engine. Such errors cannot be re-solved at the character matching level, but it can probably be matched at the syllable string level. The confusion matrix is used to further reduce the effect of speech recognition errors due

to similar sounding characters.

To account for possible errors in CSS components, we perform similarity, instead of exact, matching at the three levels. Given the basic CSS component q_i, and a phrase c_j in the dictionary, we compute:

$$Sim(q_i, c_i) = \frac{LCS(q_i, c_i)}{\max\{|q_i|, |c_i|\}} * \sum_{k=0}^{LCS(q_i,c_i)} M_k \quad (2)$$

where $LCS(q_i,c_j)$ gives the number of characters/ syllable matched between q_i and c_i in the order of their appearance using the longest common subsequence matching (LCS) algorithm (Cormen et al 1990). M_k is introduced to accounts for the similarity between the two matching units, and is dependent on the level of matching. If the matching is performed at the character or syllable string levels, the basic matching unit is one character or one syllable and the similarity between the two matching units is 1. If the matching is done at the confusion syllable string level, M_k is the corresponding coefficients in the confusion matrix. Hence LCS (q_i,c_j) gives the degree of match between q_i and c_j, normalized by the maximum length of q_i or c_j; and ΣM gives the degree of similarity between the units being matched.

The three level of matching also ranges from being more exact at the character level, to less exact at the confusion syllable level. Thus if we can find a relevant phrase with $sim(q_i,c_j)>\delta$ at the higher character level, we will not perform further matching at the lower levels. Otherwise, we will relax the constraint to perform the matching at successively lower levels, probably at the expense of precision.

The detail of algorithm is listed as follows:

Input: Basic CSS Component, q_i

a. Match q_i with phrases in dictionary at character level using Eqn.(2).
b. If we cannot find a match, then match q_i with phrases at the syllable level using Eqn.(2).
c. If we still cannot find a match, match q_i with phrases at the confusion syllable level using Eqn.(2).
d. If we found a match, set $q'_i=c_j$; otherwise set $q'_i=q_i$.

For example, given a query: "请问有什么关于伊拉克的新闻" (please tell me some news about Iraq). If the query is wrongly recognized as "城文柳树们关于伊拉克的新闻". If, however, we could correctly extract the CSS "伊拉克" (Iraq) from this mis-recognized query, then we could ignore the speech recognition errors in other parts of the above query. Even if there are errors in the CSS extracted, such as "陈(chen) 水边(waterside)" instead of "陈水扁(chen shui bian)", we could apply the syllable string level matching to correct the homophone errors. For CSS errors such as "贪(corrupt) 一般(usually)" instead of the correct CSS "塔利班(Taliban)", which could not be corrected at the syllable string matching level, we could apply the confusion syllable string matching to overcome this error.

5 Experiments and analysis

As our system aims to correct the errors and extract CSS components in spoken queries, it is important to demonstrate that our system is able to handle queries of different characteristics. To this end, we devised two sets of test queries as follows.

a) Corpus with short queries

We devised 10 queries, each containing a CSS with only one basic component. This is the typical type of queries posed by the users on the web. We asked 10 different people to "speak" the queries, and used the IBM ViaVoice 98 to perform the speech to text conversion. This gives rise to a collection of 100 spoken queries. There is a total of 1,340 Chinese characters in the test queries with a speech recognition error rate of 32.5%.

b) Corpus with long queries

In order to test on queries used in standard test corpuses, we adopted the query topics (1-10) employed in TREC-5 Chinese-Language track. Here each query contains more than one key semantic component. We rephrased the queries into natural language query format, and asked twelve subjects to "read" the queries. We again used the IBM ViaVoice 98 to perform the speech recognition on the resulting 120 different spoken queries, giving rise to a total of 2,354 Chinese characters with a speech recognition error rate of 23.75%.

We devised two experiments to evaluate the performance of our techniques. The first experiment was designed to test the effectiveness of our query model in extracting CSSs. The second was designed to test the accuracy of our overall system in extracting basic query components.

5.1 Test 1: Accuracy of extracting CSSs

The test results show that by using our query model, we could correctly extract 99% and 96% of CSSs from the spoken queries for the short and long query category respectively. The errors are mainly due to the wrong tagging of some query components, which caused the query model to miss the correct query structure, or match to a wrong structure.

For example: given the query "请问有什么关于塔利班的新闻" (please tell me some news about Taliban). If it is wrongly recognized as:

陈文流苏忙 关于 谭利班的 下午
9 7 9 10

which is a nonsensical sentence. Since the probabilities of occurrence both query structures [0,9,7] and [7,9,10] are 0, we could not find the CSS at all. This error is mainly due to the mis-recognition of the last query component "新闻 (news)" to "下午 (afternoon)". It confuses the Query Model, which could not find the correct CSS.

The overall results indicate that there are fewer errors in short queries as such queries contain only one CSS component. This is encouraging as in practice most users issue only short queries.

5.2 Test 2: Accuracy of extracting basic query components

In order to test the accuracy of extracting basic query components, we asked one subject to manually divide the CSS into basic components, and used that as the ground truth. We compared the following two methods of extracting CSS components:

a) As a baseline, we simply performed the standard stop word removal and divided the query into components with the help of a dictionary. However, there is no attempt to correct the speech recognition errors in these components. Here we assume that the natural language query is a bag of words with stop word removed (Ricardo, 1999). Currently, most search engines are based on this approach.
b) We applied our query model to extract CSS and employed the multi-tier mapping approach to extract and correct the errors in the basic CSS components.

Tables 3 and 4 give the comparisons between Methods (a) and (b), which clearly show that our method outperforms the baseline method by over 20.2% and 20 % in F_1 measure for the short and long queries respectively.

Table 3: Comparison of Methods a and b for short query

	Average Precision	Average Recall	F_1
Method a	31%	58.5%	40.5%
Method b	53.98%	69.4%	60.7%
	+22.98%	+10.9%	+20.2%

Table 4: Comparison of Methods a and b for long query

	Average Precision	Average Recall	F_1
Method a	39.23%	85.99%	53.9%
Method b	67.75%	81.31%	73.9%
	+28.52%	-4.68%	+20.0%

The improvement is largely due to the use of our approach to extract CSS and correct the speech recognition errors in the CSS components. More detailed analysis of long queries in Table 3 reveals that our method performs worse than the baseline method in recall. This is mainly due to errors in extracting and breaking CSS into basic components. Although we used the multi-tier mapping approach to reduce the errors from speech recognition, its improvement is insufficient to offset the lost in recall due to errors in extracting CSS. On the other hand, for the short query cases, without the errors in breaking CSS, our system is more effective than the baseline in recall. It is noted that in both cases, our system performs significantly better than the baseline in terms of precision and F_1 measures.

6 Conclusion

Although research on natural language query processing and speech recognition has been carried out for many years, the combination of these two approaches to help a large population of infrequent users to "surf the web by voice" has been relatively recent. This paper outlines a divide-and-conquer approach to alleviate the effect of speech recognition error, and in extracting key CSS components for use in a standard search engine to retrieve relevant documents. The main innovative steps in our system are: (a) we use a query model to isolate CSS in speech queries; (b) we break the CSS into basic components; and (c) we employ a multi-tier approach to map the basic components to known phrases in the dictionary. The tests demonstrate that our approach is effective.

The work is only the beginning. Further research can be carried out as follows. First, as most of the queries are about named entities such as the persons or organizations, we need to perform named entity analysis on the queries to better extract its structure, and in mapping to known named entities. Second, most speech recognition engine will return a list of probable words for each syllable. This could be incorporated into our framework to facilitate multi-tier mapping.

References

Berlin Chen, Hsin-min Wang, and Lin-Shan Lee (2001), "Improved Spoken Document Retrieval by Exploring Extra Acoustic and Linguistic Cues", Proceedings of the 7th European Conference on Speech Communication and Technology located at http://homepage.iis.sinica.edu.tw/

Paul S. Jacobs and Lisa F. Rau (1993), Innovations in Text Interpretation, Artificial Intelligence, Volume 63, October 1993 (Special Issue on Text Understanding) pp.143-191

Thomas H. Cormen, Charles E. Leiserson and Ronald L. Rivest (1990), "Introduction to algorithms", published by McGraw-Hill.

Jerry R. Hobbs, et al,(1997) , FASTUS: A Cascaded Finite-State Transducer for Extracting Information from Natural-Language Text, Finite-State Language Processing, Emmanuel Roche and Yves Schabes, pp. 383 - 406, MIT Press,

Julian Kupiec (1993), MURAX: "A robust linguistic approach for question answering using an one-line encyclopedia", Proceedings of 16th annual conference on Research and Development in Information Retrieval (SIGIR), pp.181-190

Chin-Hui Lee et al (1996), "A Survey on Automatic Speech Recognition with an Illustrative Example On Continuous Speech Recognition of Mandarin", in Computational Linguistics and Chinese Language Processing, pp. 1-36

Helen Meng and Pui Yu Hui (2001), "Spoken Document Retrieval for the languages of Hong Kong", International Symposium on Intelligent Multimedia, Video and Speech Processing, May 2001, located at www.se.cuhk.edu.hk/PEOPLE/

Kenney Ng (2000), "Information Fusion For Spoken Document Retrieval", Proceedings of ICASSP'00, Istanbul, Turkey, Jun, located at http://www.sls.lcs.mit.edu/sls/publications/

Hsiao Tieh Pu (2000), "Understanding Chinese Users' Information Behaviors through Analysis of Web Search Term Logs", Journal of Computers, pp.75-82

Liqin, Shen, Haixin Chai, Yong Qin and Tang Donald (1998), "Character Error Correction for Chinese Speech Recognition System", Proceedings of International Symposium on Chinese Spoken Language Processing Symposium Proceedings, pp.136-138

Amit Singhal and Fernando Pereira (1999), "Document Expansion for Speech Retrieval", Proceedings of the 22nd Annual International conference on Research and Development in Information Retrieval (SIGIR), pp. 34~41

Tomek Strzalkowski (1999), "Natural language information retrieval", Boston: Kluwer Publishing.

Gang Wang (2002), "Web surfing by Chinese Speech", Master thesis, National University of Singapore.

Hsin-min Wang, Helen Meng, Patrick Schone, Berlin Chen and Wai-Kt Lo (2001), "Multi-Scale Audio Indexing for translingual spoken document retrieval", Proceedings of IEEE International Conference on Acoustics, Speech, Signal processing , Salt Lake City, USA, May 2001, located at http://www.iis.sinica.edu.tw/~whm/

Yongcheng Wang (1992), Technology and basis of Chinese Information Processing, Shanghai Jiao Tong University Press

Baeza-Yates, Ricardo and Ribeiro-Neto, Berthier (1999), "Introduction to modern information retrieval", Published by London: Library Association Publishing.

Hai-nan Ying, Yong Ji and Wei Shen, (2002), "report of query log", internal report in Shanghai Jiao Tong University

Guodong Zhou and Kim Teng Lua (1997) Detection of Unknown Chinese Words Using a Hybrid Approach Computer Processing of Oriental Languages, Vol 11, No 1, 1997, 63-75

Guodong Zhou (1997), "Language Modelling in Mandarin Speech Recognition", Ph.D. Thesis, National University of Singapore.