A Method to Define a Search Query by Speech

Pavel Žikovský
Dept. of Computer Science and Engineering
CTU Prague
Karlovo nám. 13, 121 35 Prague 2
Czech Republic
xzikovsk@fel.cvut.cz

Pavel Slavík
Dept. of Computer Science and Engineering
CTU Prague
Karlovo nám. 13, 121 35 Prague 2
Czech Republic
slavik@cs.felk.cvut.cz

Abstract. Making the content of web pages more accessible for visually impaired people involves displaying web pages sonically - mostly by speech. The other aspect of web browsing - entering information into web forms remains nearly untouched. This aspect is one of the biggest advantages of the internet itself - the possibility to use search engines - Google, etc. Before searching the Internet, the user must enter a query, which is composed of a list of isolated keywords. To recognize a spoken query for a search engine, it is necessary to recognize a linear list of keywords separately. As these can be nearly any combination of words or abbreviations, the recognition is a difficult task for any current speech recognizer. In this paper, we describe an efficient method to enter a query by speech. Our method is based on predicting a list of keywords for the next word (which is the user going to dictate) from precedent keywords. The fact, that as a result of this prediction there is only a limited set of keywords to choose from, dramatically increases the precision of the recognizer. The algorithm itself is based on cluster analysis and is applicable for virtually any database query.

Keywords: Speech, Internet, searching process, query, keywords, cluster analysis

1 Introduction

The possibility to search is one of the biggest advantages of the Internet itself. Without this possibility, the Internet would be like completely unorganized library with billions of titles - in other words - nearly useless. Fortunately, there are many search engines on the Internet (Google, Yahoo, Altavista,...) and all of them work on a similar principle - user enters a query composed of separate keywords and submits the query. This scheme is fine for most users, but for visually impaired people the current speech recognizers are not able to recognize a list of isolated keywords with sufficient success. Current approaches to define a search query by speech mostly consider the query to be a normal utterance. This consideration makes it difficult to recognize the query, because it does follow the structure of common utterance at all.

When searching any database (Internet search engines are special case of databases) we usually type in keywords, which we want to find. In speech-based interfaces, this necessitates the recognition of user utterance, which contains the query. The problem is that technology fir the recognition of continuous speech is still undeveloped. Even if many approaches have

been made, recognition of continuous speech, which would be able to understand many different voices with satisfactory error rate, still not exists. Fortunately, the database query follows some rules and patterns we can utilize to achieve efficient recognition. Most of all - the query is a linear list of keywords. This list of separated keywords leads to another algorithm - the recognition of isolated words. The precision of isolated spoken word recognition relies mainly on the number of possible words the recognizer chooses from its dictionary. The precision of recognizer follows a simple rule: As the number of possibilities in a given situation decreases, the precision of the recognizer increases [10]. To achieve the best recognition rate, the dictionary should be as small as possible (or at least limited). The key concept within this paper is to predict the next keyword in a real-time query while user dictates it. This prediction allows us to limit the dictionary for each keyword in the query (except the first one).

Our prediction algorithm is based on cluster analysis. Cluster analysis is an exploratory data analysis algorithm (or algorithms) for solving classification problems. Its object is to sort cases (things, events, etc) into groups, or clusters, so that the degree of association is strong among members of the same cluster and weak among members of different clusters. [1,5,7] Each cluster describes the class to which its members belong. In our case, we are trying to find clusters of keywords, which are the most probable in the meaning of refining the results of previous query. Each cluster is represented by one predicted keyword. From these clusters, we choose those with the highest probability rate. This set of keywords (clusters) represents the list of keywords to predict. Of course this list is not complete because user might want to give completely different keyword, which is not in our list, but in the meaning of prediction success rate we still achieve quite good results. In the case, where the user did not find the desired word within our list, the situation is similar as with the first keyword - it is completely unknown what the user is going to dictate, so there is no chance to predict. The overall scheme of such a system is illustrated in Figure 1.

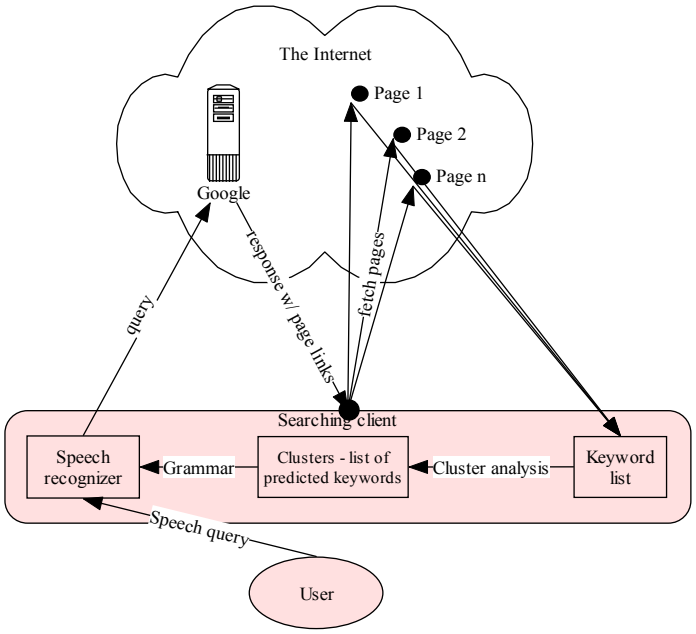

Figure 1 - Overall System Scheme

To implement our algorithm, we used Google API[1] [9], which is an application interface to the Google search engine based on SOAP[2] technology. During the testing stage, which was tested using visual model (see section 3), we also realized that the algorithm is useful as a universal "searching assistant" tool for any modality - there are moments when we do not know which keyword should be added to our query to find the word we are searching for. This most often happens when we are not sure about what are we searching for.
The tests and their results are outlined in the section "Tests", meanwhile the algorithm itself is described in the following section.

2 The Algorithm

Each web (HTML) page on the Internet which is indexed by some search engine has a list of keywords affiliated with it. In HTML, they are usually defined in so-called "meta tags", which are directly embedded in pages. These keywords are used to index the page by a search engine; therefore all of the resulting pages of a query must contain these keywords. In other words, each query corresponds to a list of pages (or any data) which fulfills its criteria. With all the pages loaded, we can perform a cluster analysis [5] of the keywords from all the pages. It means that we create a list of keywords for each page which is the result of the previous query, and within this list we find keywords with the highest occurrence rate. The basic scheme of the process is shown in Figure 2.

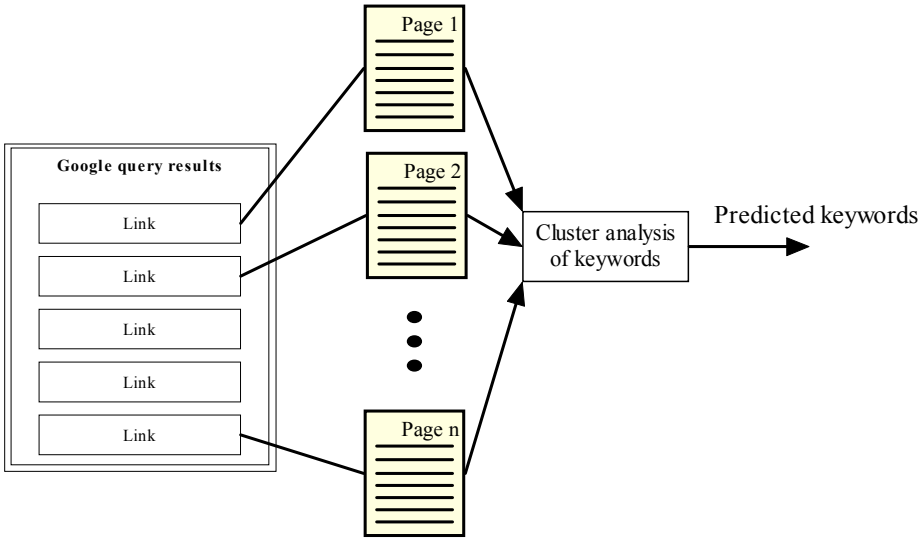

Figure 2 - Basic keyword prediction scheme

What follows is an example of the keyword searching process. For explanation purposes, we will not think about speech right now. The first keyword we enter is *music*. We perform a query with this keyword to the Google search engine [9]. It loads the first 20 results (this value is adjustable and it is set to 20 for purposes of speed) and analyzes the set of keywords on each page. The analyzing process scans the keywords contained within these 20 pages and

[1] API - Application Interface
[2] Communication protocol based on XML

tries to locate a keyword, whose occurrence is multiple within the given pages. In our example, it found keywords *mp3*, *free, band* and *downloads*. These four keywords exceeded the minimal boundary to be displayed. In our case, this value was set to 3. There can be other strategies to pick the best keys, such as pick first *n* keywords with the highest occurrence rate. Detailed information about the keywords we found and their occurrence count follows in Table 1.

Table 1: Keyword count: Cluster analysis results of the query "music"

Keyword	mp3	free	band	downloads
Occurrence count (#pages)	6	4	4	3

In our example, "music" is the first keyword and we assume that the user is going to say one of the words we have found. We add the keyword user chose or said to the query (so now it is i.e. "music free") and the algorithm of predicting next keyword starts in exactly the same manner as with the "music" case. Figure 3 bellow, shows a scheme of how users move within the space of keywords, which define the query. So after choosing the second keyword ("free"), the user is again given a list of keywords to refine the query. The complete trajectory of the user within the keyword space is marked with thick lines. As the complexity (keyword count) of the query increases, the number of resulting records decreases.
There is also a slight chance to simply predict even the first keyword - it can be a list of the most frequently used keywords. As there are many of them, and their structure is nearly identical to the main categories on search engines, there is only a slight chance that user's first keyword will fit into this set of keywords.

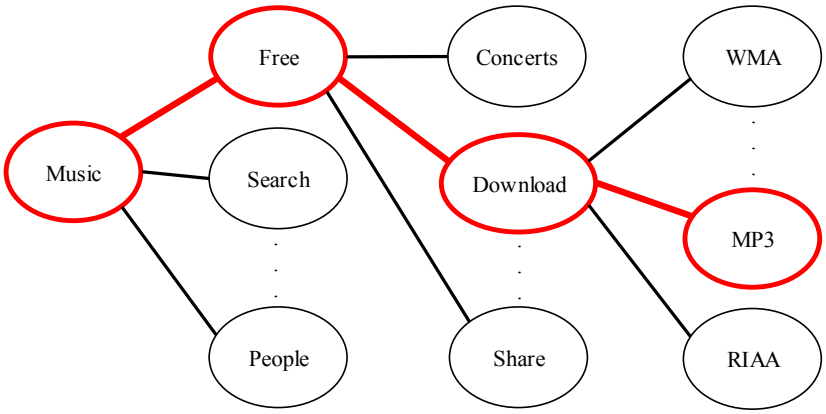

Figure 3 - Way through keyword space

3 Visual Interface - Test Setup

For testing purposes, we have also created a visual display for the algorithm. We take this step because we were not able to find more than one visually impaired man at that moment. Because we are testing the efficiency of our keyword prediction algorithm and not any particular speech recognizer, we can use the visual interface without losing the general purpose. During the testing stage we also realized, that the method presented here is very

usable for sighted users. The visual display is displayed in Figure 4. We have chosen such a "planetary" model, because we feel it is capable of displaying the structure of keywords very well. The situation on Figure 4 is exactly the same as in the first column on Figure 3. The user just entered the first keyword (music) and is given a list of the next possible keywords (free, people, search ...)

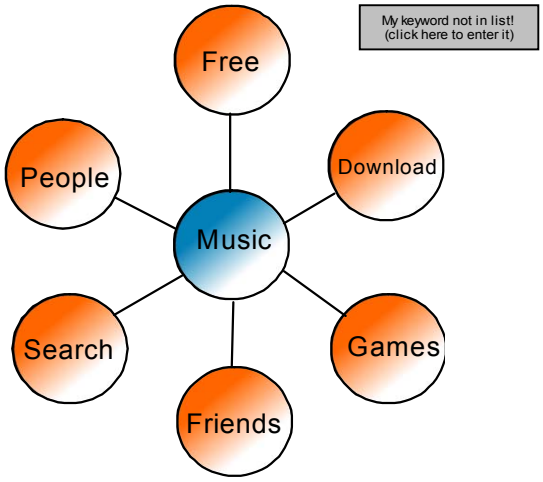

Figure 4 - Visual interface: "found keyword" structure

This testing interface works as follows: As the user enters first keyword, we perform a cluster analysis of the resulting information (pages), which generates list of next possible keywords. These new keywords, which have the capacity to refine the query in the best possible way, are displayed as planets in a planetary system with the initial keyword (or a set of keywords; "music" in our example) in the center (similar to a sun). Clicking the sun results in opening the corresponding Google page. Clicking orbital balls will refine the central query with its keyword and the program will perform another cluster analysis. In other words, in the central ball, there is the current query and the orbital balls represent keywords which should define the query in the center.

For the case the prediction is not suitable - the desired keyword is not in list - there is a box to enter it manually (see the gray rectangle in the top-right of Figure 3). This box also flashes a message if the search yields no results.

The pilot version of the speech enabled system is based on Microsoft speech API [6]. As a input for isolated word recognition, we used speech recognition grammar. In our case, the grammar is a simple XML-based document with a list of keywords to recognize inside [6].

4 Tests

We conducted two research tests concerning the efficiency of the keyword prediction procedure on our testing prototype. In the first test, we measured the number of unsuccessful predictions (i.e. when the desired keyword was not within the set we have predicted) and its relation to query length. The following tables show a sample of queries we used for testing.

Table 2: Successfully predicted keywords

php SOAP library "open source"	
keyword	**Sucessfully predicted?**
php	N/A
SOAP	NO
library	YES
open source	YES

delphi components freeware graphics	
keyword	**Sucessfully predicted?**
delphi	N/A
components	YES
freeware	YES
graphics	YES

TTS Windows downloads freeware	
keyword	**Sucessfully predicted?**
TTS	N/A
Windows	NO
downloads	YES
freeware	YES

VoiceXML server downloads free	
keyword	**Sucessfully predicted?**
VoiceXML	N/A
server	YES
downloads	YES
free	YES

We concluded 16 tests with various keyword counts (query length).
We also looked for any dependency between the query length and percentage of query drop-out (Table 3). The test proves that more complex queries result in better prediction.

Table 3: Keyword drop-out count

Keyword count	2	3	4	5
% next keyword drop-out	32	28	21	26

In the second test, subjects were asked to find a certain page or information (see Table 4) using "native query" and query composed from elements of the keyword list provided by our algorithm. The queries were taken as “similar” if they contained the same keyword count and when the page we were looking for was a result of all the keywords (see Table 4 for examples). The result is the average number of user interactions while refining the query (mouse clicks and typing).

Table 4: Results of the efficiency tests comparing two searching methods

Page (task) description	Desired query	Iteration count (Google)	Iteration count (with query prediction)
Find prices of tamagochi toys in Oregon	+tamagochi+Oregon+shop +online	12	6
Find freeware component for parsing AltaVista results for Borland Delphi environment	+Delphi+component+ freeware+altavista+parsing	14	8
Find freeware plug-in for Adobe Photoshop, which performs charcoal effect	+photoshop +plugin+freeware+charcoal	8	3
Find 3D model of some Le Corbusier’s villa	+3D +model+LeCorbusier+villa	9	5

The results of our test proved the efficiency of our keyword prediction method. The result of the first test, with average 25 percent of query drop-out (75% success), is quite good. The

second test shows that the increase of searching efficiency also is not neglible - the gain of the searching process speed was almost 200 percent!
The speech tests, which were made using Microsoft SAPI (see section 3), have been done in the same manner as visual tests with and the same results.

5 Conclusions and Future Work

We have presented a novel method to help visually impaired Internet user to search the internet more easily and efficiently. The help we provide is based on cluster analysis and next-keyword query prediction. The output of such a prediction is a list of keywords, which are the most probable candidates to extend the query. We use these keywords to limit the vocabulary size for the speech recognizer. Test we have performed assured us about potential benefits of our work for visually impaired users.
We would like to implement (we are working on it) a working prototype of our algorithm together with a speech recognition engine to prove our contribution to speech-driven web search. The application is about to be VoiceXML web-based application and its overall scheme is illustrated on the following figure. So far, we have made only a pilot version for testing purposes. We would also like to implement the tracking and saving of keyword groups for each user. This way we will be able to offer “what other people choose” options for each user to optimize the keyword list.
During research we encountered one problem: Not all pages have their keywords in "<meta>" tag. It would be ideal if Google API provides the list of keywords using SOAP. The fact that we have to fetch all resulting pages to find keywords slows down the searching process.

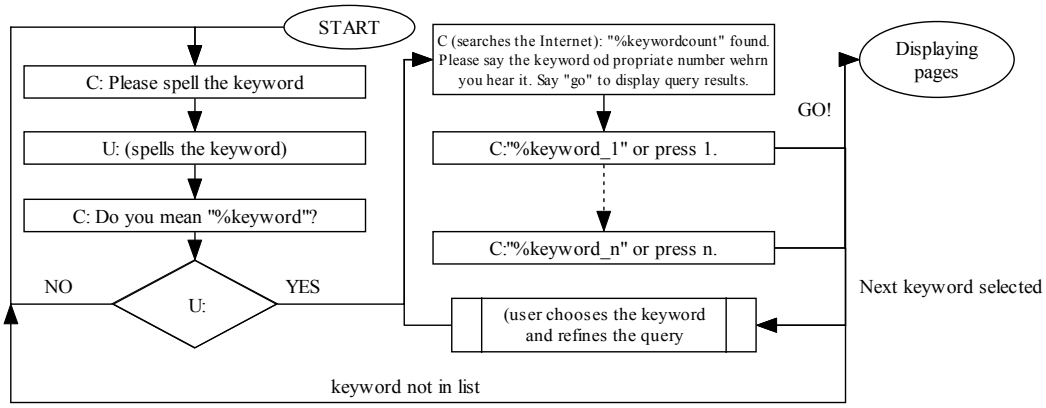

Figure 5 - Speech enabled system scheme

6 Acknowledgements

This project has been partially supported by the Ministry of Education, Youth and Sports of the Czech Republic under research program No. Y04/98: 212300014 (Research in the area of information technologies and communications) and research program GACR 201/02/1553.

7 References

[1] Andersen, E. ,1992. *The Statistical Analysis of Categorical Data*. Springer-Verlag London Ltd.

[2] Jelinek, F., 1997. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA.

[3] Manning, C. and Schutze, H. 1999. *Foundations of Statistical Natural Language Processing.* MIT Press, Cambridge, MA.

[4] Dufresne, C. et al., 1995. *Multimodal user interface system for blind and "visually occupied" users: Ergonomic evaluation of the haptic and auditive dimensions.* In Proceedings of IFIP International Conference Interactions '95 Lillehammer, Norway. (pp. 163-168).

[5] Kaski S., 1997. *Data Exploration Using Self-Organizing Maps*. Acta Polytechnica Scandinavica.

[6] *Microsoft Speech Application Program Interface (SAPI)* Version 5.0, http://www.microsoft.com/speech.

[7] A.Leuski., 2001. *Evaluating document clustering for interactive information retrieval.* In Proceedings of the ACM CIKM 2001 Tenth International Conference on Information and Knowledge Management, pp. 33, 40.

[8] Joachims T., 2002. *Optimizing Search Engines Using Clickthrough Data.* Proceedings of ACM Conference on Knowledge Discovery and Data Mining, 2002.

[9] Google API, http://www.google.com/apis

[10] Becchetti C., Ricotti L. P., 1999. *Speech Recognition : Theory and C++ Implementation.* John Wiley & Sons