Improved Voice Search With Query Reformulations

Group 9: Hanying Huang, Tomas Robinson, Yu Wang

Abstract

We report on the implementation of a system for handling and resolving users input errors. We redesigned the front end of voice search to allow users to find information more quickly and to allow them to change or modify their queries with ease. We did this by implementing 4 different techniques. We list the other possible queries the voice API heard. The system also displays synonyms of the key words of the query. It then also displays homonyms of the query to allow the user to modify it without retyping. Finally the system displays related words and phrases that the user may be looking for. This implementation is a first step into having better voice search systems that are more intuitive and more successful at dealing with users inputs. We then took this system and had a few participants use and answer some questions about. In this paper we discuss the rationale behind the system, the implementation of it, and the different tools used to reach our goal of a more user-friendly system for voice search.

Keywords

Voice search, voice input errors, query reformulation

1 Introduction

1.1 Background

Mobile computing has spurred the use and research into voice search systems for everything from the standard Google search to sending text messages from the car without taking your eyes off of the road. By far, the most popular voice search implementations are Google Voice Search and Siri from Apple. People have begun to use these systems not just in the controlled environments of the lab but also out in the world. Voice search is included in many applications and it a central feature to all three of the major smartphone platforms, iOS, Android, and Windows Phone's newly introduced Cortana voice assistant. Voice search is also part of Mac OS X and Google's Chrome Browser. These implementations and others excel at certain aspects of voice search.

Our recent project aims to improve the interface and use of voice search. Previous work [1] showed that users interactions with Google Voice Search, encountered certain classes of errors that are particular to voice search. The chart below from Jiang et. al. [1] demonstrates the most common problems users had with Google Voice Search.

Table 3. Categorization of 20 words with the highest recognition error rates.

Type	Examples (# NOT recognized correctly / # used)
Acronym	ER(29/29), AVP(11/11), US(57/61), USA(6/11)
Named Entity	Owen(25/26), Culpeper(18/27), Ralph(22/36), Gulf(13/24), Falkland(14/27)
Non-English	Nino(31/46)
Other words	theft(14/14), achievement(10/10), taxing(18/21), fraud(12/14), violence(19/27), talk(9/15), sun(24/41), aspirin(23/43), embezzlement(9/18), maglev(8/16)

1.2 Problem Statement

How can a voice search system take in queries quickly while still allowing the users to easily reformulate them when the system is unable to generate the correct query or when the user is unsure about what they are searching for? We would like to allow the user to quickly reformulate queries. We do this through the use of different techniques for helping the user. The system displays the results of what it interpreted the user saying. It then shows words that may sound like the query. The system will also show phrases that are similar to the top-rated query. Our proposed system should be able to allow users to search by voice in a way that makes it easier, faster, and more accurate.

2 System Overview

In this project, we implemented a simple demo system to illustrate the possible methods to solve the stated problems in section 1.2 and also to reduce the user's overall efforts to correct their IE errors. Ideally, the system will enable the user to correct the query without typing. The figure 2.1 provides a basic feature structure about the system.

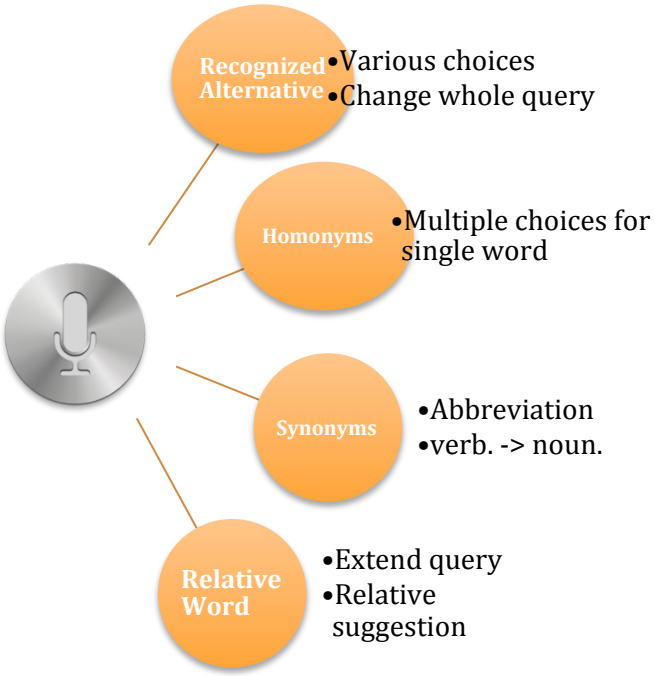


Figure 2.1 Feature Structure of the System

- **Features**

As is shown in Figure 2.1, the system has the following features:

- Basic Voice Search
- Providing Recognized Alternatives
- Providing Homonyms Suggestions
- Providing Synonyms Suggestions
- Providing Relative Words Suggestions

The details of each feature will be explained in Section 3.2, 4.1 and 4.2.

- **Interface**

We designed our interface carefully so that we can provide the user a clear, easy to use, and satisfactory interface. And the goal of the design is to let the user find the needed suggestions or alternatives easily. The details of the design are discussed in Section 3.3 and 5.

3 User Requirements

3.1 Target Users

The target users of our system are those who prefer searching with voice to typing the query directly. For instance, there are users who may be unable to type or be too busy doing other things. Moreover, the system tries to provide more support to people who are not native English speakers and have an accent.

3.2 Expected Features

As is stated in section 1.1, there are some problems in voice search, such as the abbreviation, and similar pronunciations. In order to improve the efficiency of the voice search and to reduce the user's overall efforts to correct the IE errors, the system are expected to provide the features to resolve such problems. The problems and expected strategies are shown in table 3.1.

Table 3.1 Problems and Expected Strategies

Problem	Strategy
Little Input (Just one word)	Provide relative suggestions
Abbreviation	Get synonyms for the query
Recognition incorrectly	Provide multiple recognition results
Similar pronunciations	Provide similar words

The details of the features are explained in section 4.1.

3.3 Interface Requirements

The goal of our interface design is to make a concise, comfortable, and practical interface. It should have an intuitive layout and a human-centered design. The most challenging part is how to arrange the position of alternatives, relatives, homonyms, and synonyms, so that users can easily find the exact link or button when they want to reformulate their query.

4 Conceptual Design

In this section, the features of the system are explained in detail. Beyond that, the approaches for implementing such features are stated as well. For instance, various APIs are used to get corresponding responses and we also use a MySQL database to act as a dictionary for the homonyms.

4.1 Features

In order to reduce the user's effort in correcting their errors in voice search, we designed four components to support such goals, including alternative queries, relative words, synonyms and homonyms.

- **Alternative queries**

The most important part of the voice search system is to recognize what the user said. However, since people have different accents and the current voice recognition technology is not stable, the final outcome might not be the most satisfying one. For example, when we are using Google voice search, it automatically uses one final outcome, which sometimes is not the one we want.

Realizing the above problem, we designed our interface to provide more alternative queries that are based on the user's input voice. For each input voice query, the system will provide the top 5 recognition results (including the final result) ranked in the order it believes.

- **Homonyms**

As is known to us, there are lots of homonyms in English, such as "their" and "there". It's not enough to just provide the alternative queries to substitute the whole query. So we came up with the idea of Homonym list. In short, this list will enable the user to change one term in the query in case that there is no satisfactory alternative queries. The words in the list are homonyms of the terms in the query. For instance, if the query contains ***by***, the list should look like ***by buy***. And the user can change it into ***buy*** by just clicking on the alternative word.

- **Synonyms**

It's commonly seen that different words may have the same meaning and sometimes the same abbreviation can be linked to different full names or to the alternate ways people refer to real world entities like product, people, locations and so on. The following examples are cited from Microsoft website [8].

- ✓ Product synonyms: "canon rebel t3i" is a synonym of the product Canon 600d
- ✓ People synonyms: "jlo" is a synonym of the person "Jennifer Lopez"
- ✓ Location synonyms: "sea tac" is a synonym of the location "Seattle Tacoma

International Airport"

Figuring out this problem, we want to provide the user a way to change their query easily instead of manually typing. For instance, at first the user inputs “jlo” for search. After having a look at the results, he or she wants to change the query to full name for better results. In this case, he can find "Jennifer Lopez" in the list and click on the preferred synonyms to get the new query.

- **Relative words**

Sometimes, the user will just input one word for their search. In this case, it’s difficult for narrow the search result. In these circumstances, the interface would provide relative words to the query so that the user can get some suggestions to narrow their search result. For instance, the user inputs JavaScript, the system will provide a list of relative words, such as Java, CSS, Ajax, and so on. Besides this, the user can have more flexible choices to change the query by the relative word or append the relative word to the current query.

4.2 Approaches

4.2.1 Overview

Figure 4.1 shows structure and approaches to implement the features.

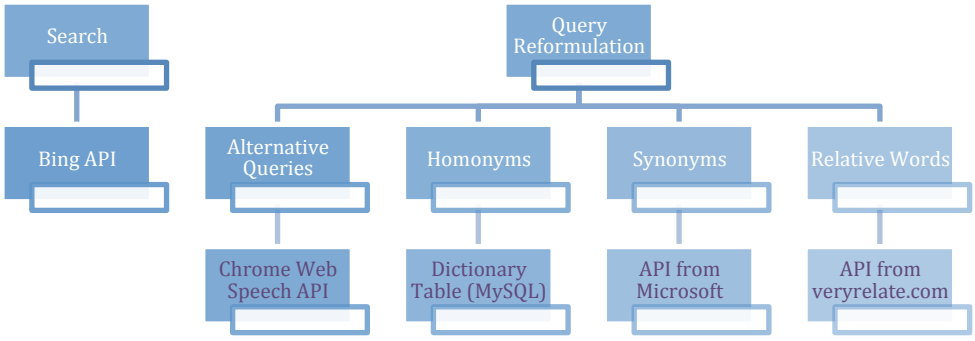


Figure 4.1 Overview of the approaches

4.2.2 Alternative Queries

There is a web speech API provided by Chrome to transfer the voice into text. By using this API, we can get a list of recognized results. This API can only be accessed through JavaScript. Moreover, we set parameter "*maxAlternatives"=5* to retrieve the first 5 results. The outcome of alternative queries is shown in Figure 4.1.

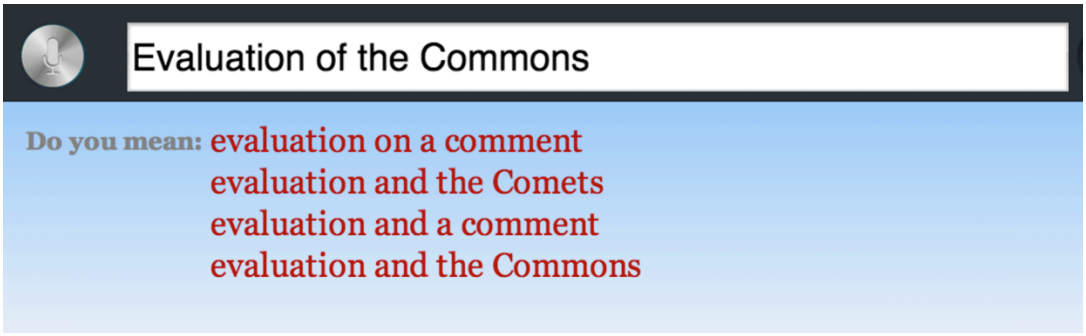


Figure 4.2 Alternative Queries

4.2.3 Homonyms

a) Database Design

The implementation of homonyms list requires a dictionary of the base word and the corresponding homonyms. The schema of the database is as followings.

Table 6.1 terms

Columns	Data Type	Properties
id	int(11)	PK AI
word	varchar(45)	
search_freq	int(11)	

- **word:** the term itself
- **search_freq:** the total count of the search frequency. To simplify the demonstration of the system, we set some fixed value to each word. The values don't have any specific meaning.

Table 6.2 similar_words

Columns	Data Type	Properties
base_id	int(11)	PK
similar_id	int(11)	PK

- **base_id:** the id of the base word.
- **similar_id:** the id of the word which is similar to the base word with base_id.

b) Algorithms

There is not a complicated algorithm to retrieve the similar words. Rather, when the similar words of the base word are retrieved, the first 5 words with higher search frequency will be returned as the final result. So we use the following query to get the required records from database.

```
$query = "select word from terms t
        where t.id IN
        (select similar_id from similar_words sw
                where sw.base_id IN (select temp.id from terms temp where temp.word='$term')
        )
order by t.search_freq desc limit 5;";
```

Figure 4.3 Query to retrieve similar words

4.2.4 Synonyms

In our system, we use the Synonym API from Microsoft to get the synonyms of the word. The system will display the top 5 results.

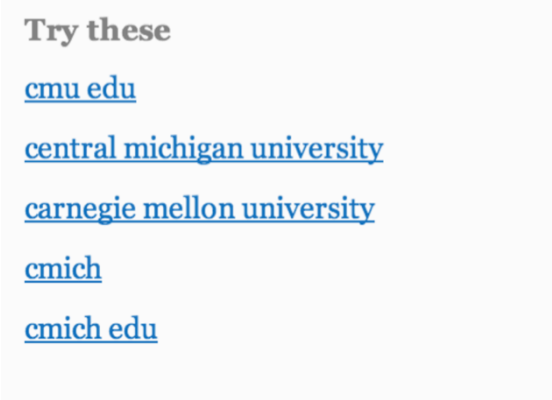


Figure 4.4 Synonyms List

4.2.5 Relative Words

By searching on various API on providing relative words, we found the one which is free and useful. This API is from veryrelated.com. The result returned is in xml format as shown in Figure 4.5 [4].

```
<Result>
  <Text>Recipe</Text>
  <HowRelated>0.690411391191 32</HowRelated>
  <Popularity>617000000</Popularity>
</Result>
```

Figure 4.5 API response in xml format

Text: The matching word's text.
HowRelated: A normalized value indicating the bidirectional relatedness of the matching word to the search term. A greater number means the words are more related. A negative number means the words are not related.
Popularity: The approximate number of pages on the Internet (in the search index) that contain the matching Text.

And as is shown in Figure 4.6, the system displays the top 6 results that are ranking by the parameter HowRelated in descending order.

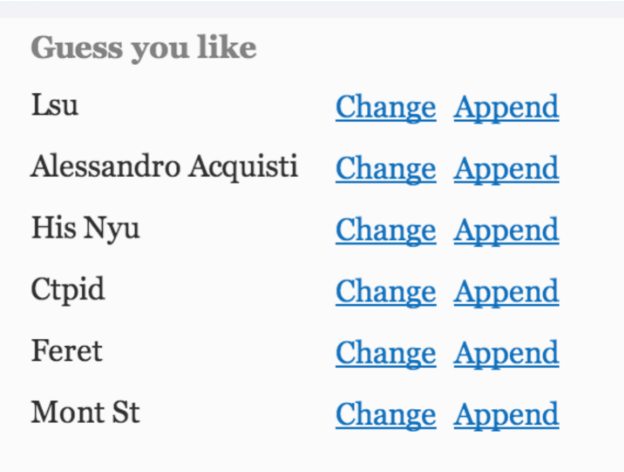


Figure 4.6 Relative Words List

5 Interface Design

5.1 Start Page interface design

The first thing we considered in the design of the interface is how many webpages do we need? To make the search engine as simple as possible, it seems that only one search page is enough; every manipulation will be done and shown on this page. However, in the end we decided to use a start page for users to initiate their search. The reason is easy. At first, 90% of the search page is empty because there are no results yet. Using a start page can make the system more complete, as well as more attractive.

For ease-of-use, there are only two elements in the start page: a big voice button, and a sentence right below the button. The sentence is used for a prompt before speaking as well as displaying the result after speaking.

Figure 5.1 Start page after speaking

5.2 Search page design

5.2.1 Reformulation Widgets Design

After clicking the button again, the page jumps to the search page, which shows the search results and all the query reformulation units. Each search result consists of 3 parts: title, description, and webpage link. Reformulation units include recognized alternatives, relatives, synonyms, and homonyms.

Recognized alternatives are a list of ranking results after the voice system test the user's speaking. It lists several of other possible recognition. Thus the length of each item is basically same as the query. Considering sometimes long query needed, we display one alternative in each line, and at most five items.

Figure 5.2 Alternative Queries

Relatives are something related to the current query. It helps the user to reformulate a more useful query. Sometimes users want to append a relative item after the current query, whereas sometimes they just want to replace it. Thus, we designed two links for each item,

so that users can choose either append or replace. The relatives list contains at most 6 items.

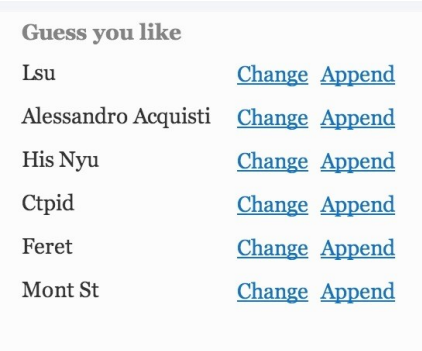


Figure 5.3 Relative Words List

Synonyms provide chances for users to reformulate their query with the same meaning but different words. Each item is just a link itself, so that users click what they want. The list shows at most 5 items.

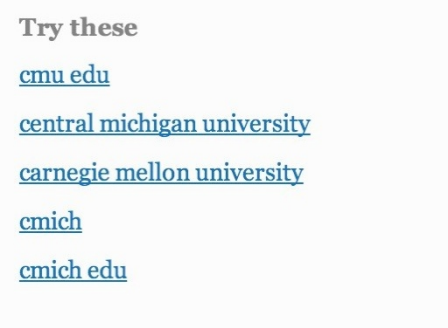


Figure 5.4 Synonyms List

Homonyms use to replace some specific words that the system got wrong during voice detection. The format of each item should be one word followed by some homonyms links.

Figure 5.5 Homonyms List

5.2.2 Reformulation Widgets Arrangement

A challenging part of this interface design was to arrange the position of these reformulation units. Considering the recognized alternatives could be long sentences, and users are much more likely to use them right after their speaking, we put the alternatives

right below the search bar (webpage header). The other three parts are arranged to the right of the page. In addition, users will know if there are some incorrect homonym words in the query right after the query formed without browsing the result, thereby the homonyms units should be on the top right in order to make it easy to be seen at the first time. Then relatives should be followed. Users may compare the results description with their information need, as well as the relatives. At this time, they maybe more interested in a relative. Last, the synonyms unit should be at the bottom right of the page, because users want to try a synonym only after they are not satisfy to all the results. At that time, the user more likely to have scrolled down the search page.

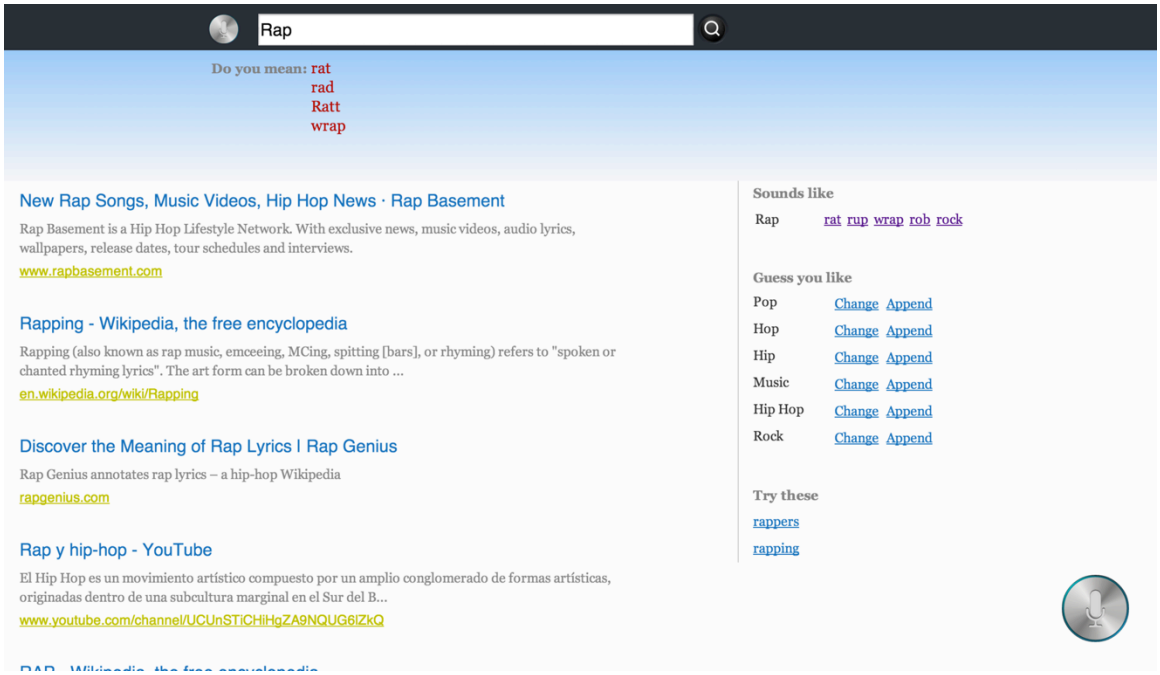


Figure 5.6 Search Page with All Widgets

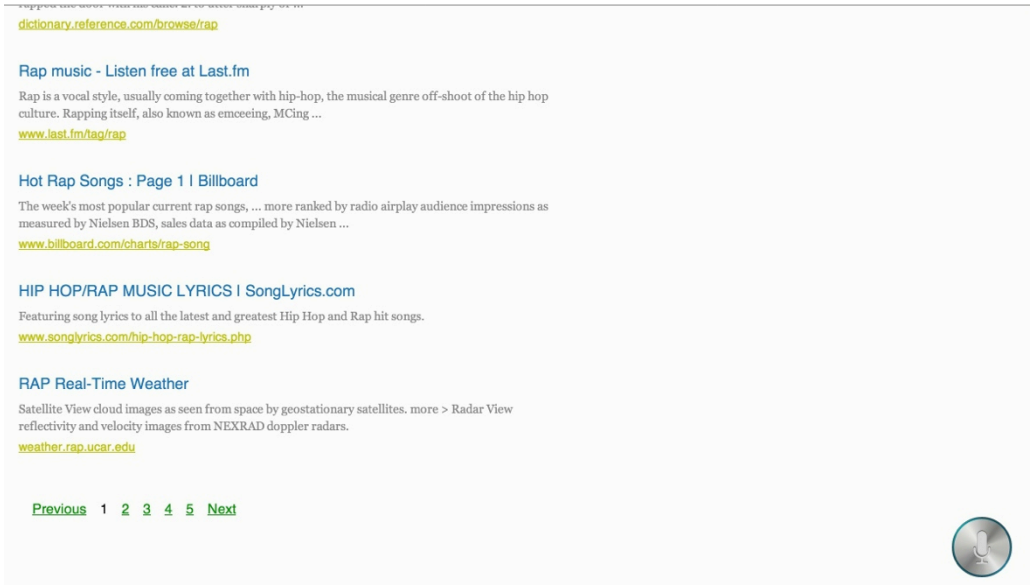
To enhance the user experience, we try to avoid using words that not all of users know what they really refer to, such as alternatives, relatives and so on. Instead, we use “Do you mean” to replace for alternatives, “sounds like” for homonyms, “Guess you like” for relatives, and “Try these” for synonyms.

5.3 Search bar design and voice icon utilization

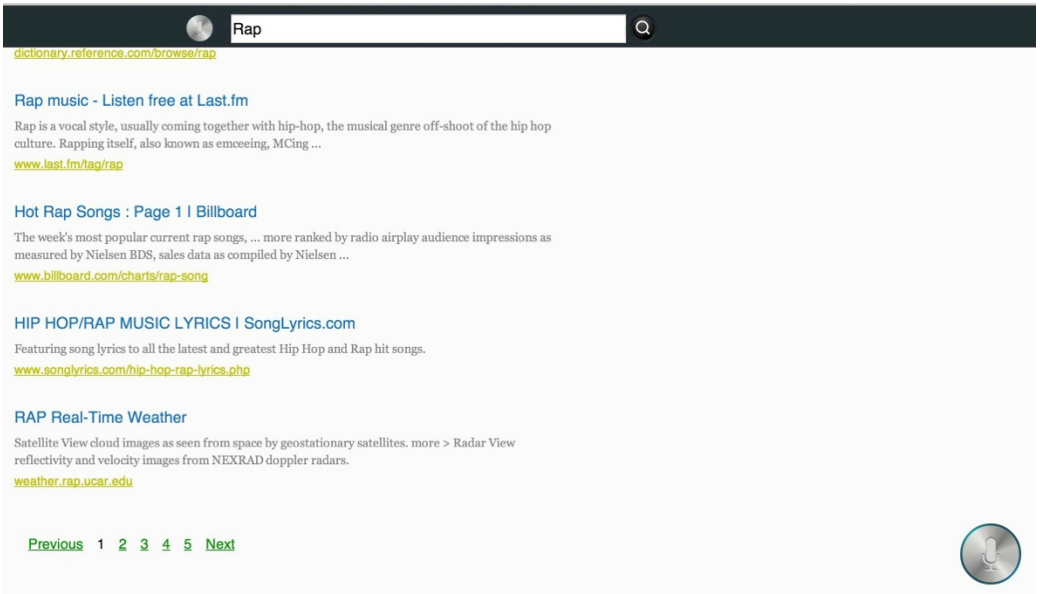
One good innovation we have made is the search bar. We put the search bar into a moving header. When users scroll done to browse the result items, the header will slide up and be hidden. Anytime users scroll up a bit, the header will slide out again. This has many benefits. First, it makes the webpage more concise, enabling more space to display the search results. Second, every time users scroll up, it may be because they want to go to the top and change query. This design will provide more convenience for users. What’s more, based on this design, we don’t need an additional search bar locate at the bottom of the page, which is the way that most of the traditional search engine did. To make the interface even more

convenient, we also create a button and fix it at the bottom right of the webpage, thereby anytime users want to speak to create a new query, they can just click this button.

- Scroll down, the header hide automatically.



- When scroll up a little bit, the header and the search bar slide out again.



6 Sample Scenarios

In this section, we will give some sample scenarios where the system would be used.

6.1 Scenario 1

✓ **User**

A woman who is cooking

✓ **Specialty**

It's a common situation that we want to look for some tips or recipes when we are cooking. In the past, the user would check cookbooks. However, it's much easier and more efficient to search online now. Additionally, when we are cooking, is there a way to avoid washing your hands again and again before typing on the keyboard? The answer is to use our system and one finger is enough.

✓ **Goal**

Find a recipe with cabbage

✓ **Search Process and Result**

1. The woman wants to find some tips for cooking with potatoes and says "cabbage".

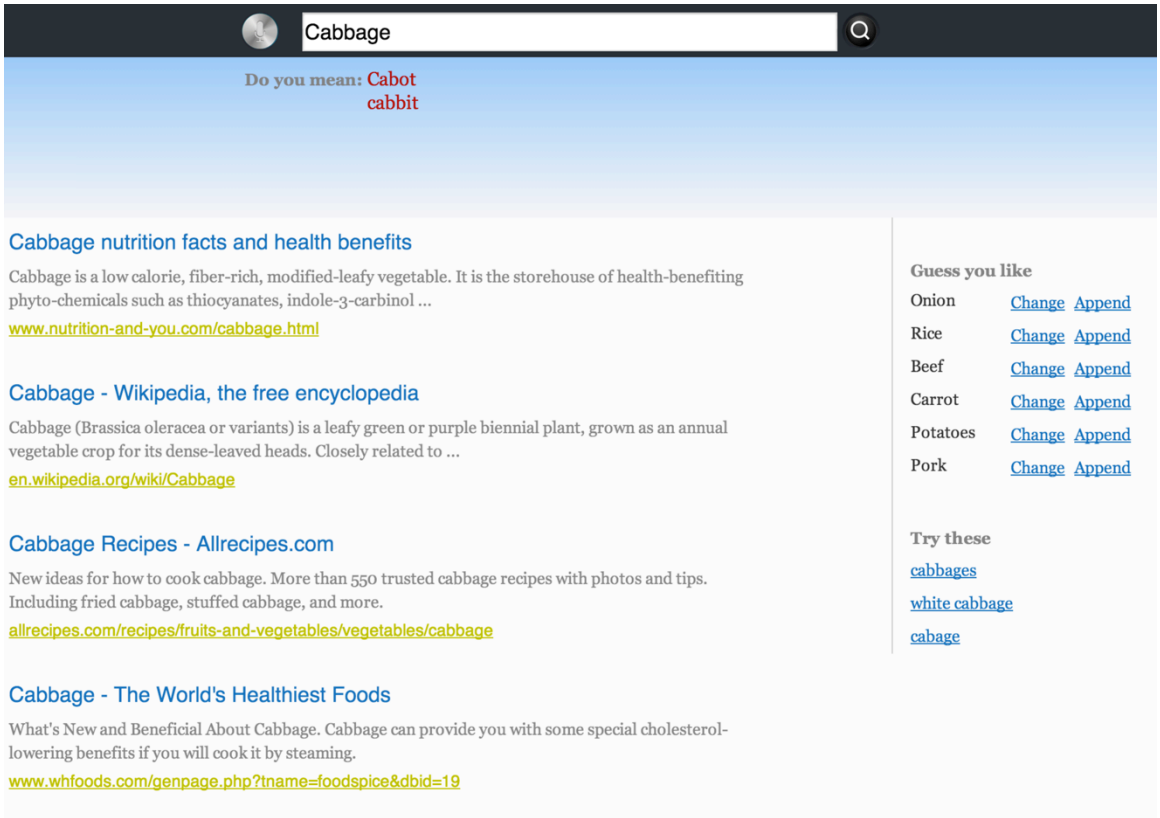


Figure 6.1

2. Review the suggestion list
3. She is interested in cooking cabbage and beef. So click on the "append" after "Beef" at the right side.

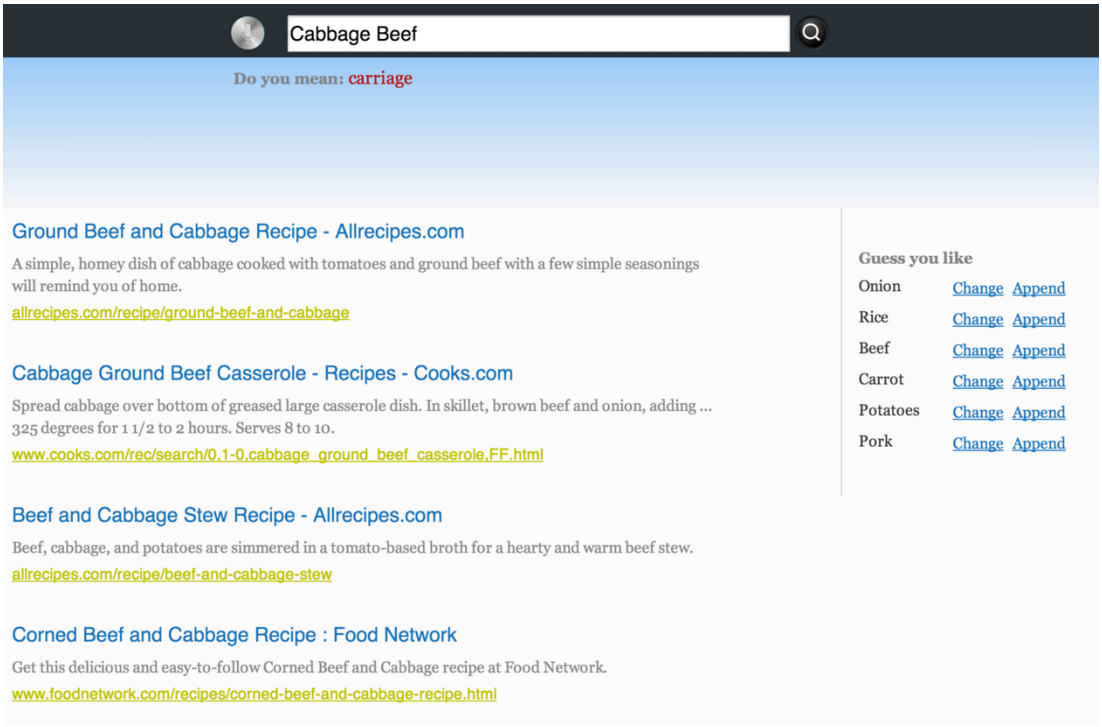


Figure 6.2

4. Find the recipe

Figure 6.3

6.2 Scenario 2

✓ **User**

The patient who has difficulties in typing

✓ **Specialty**

The user might be temporarily or permanently disabled. In this case, he or she is not able to type the query.

✓ **Goal**

Search news or articles about NCAA and basketball champions.

✓ **Search Process and Result**

1. The patient says “N-C-A-A” for searching

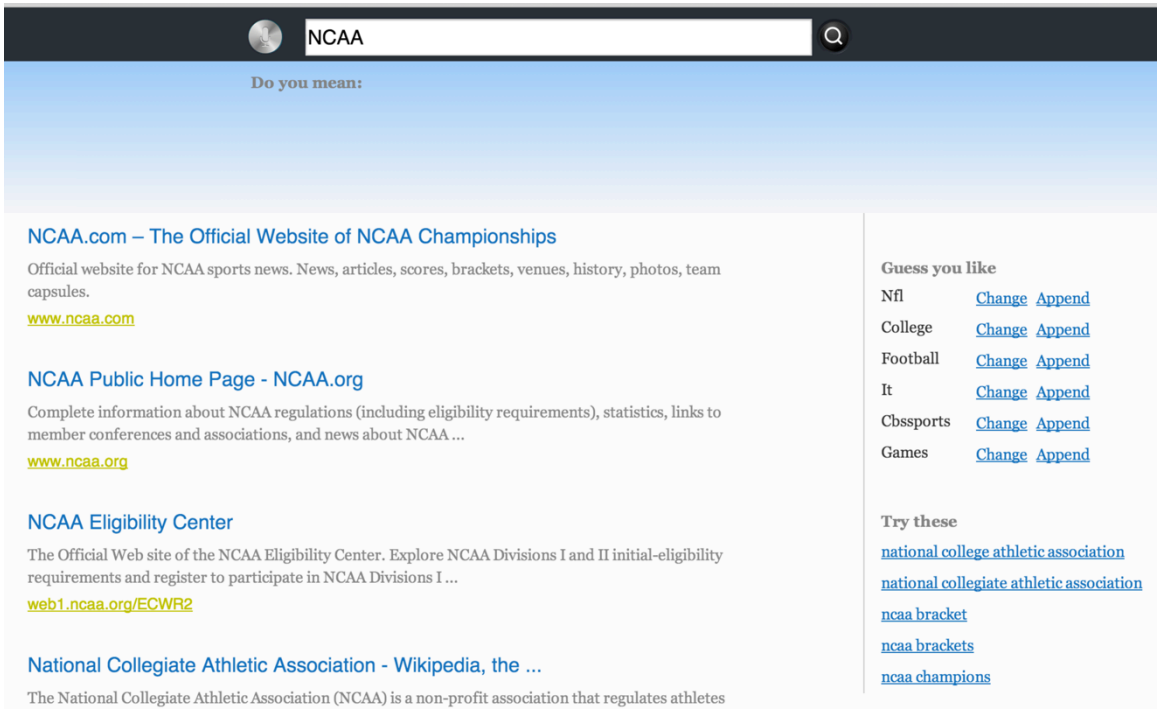


Figure 6.4

2. Change the query to NCAA champions by clicking the one under “Try these”

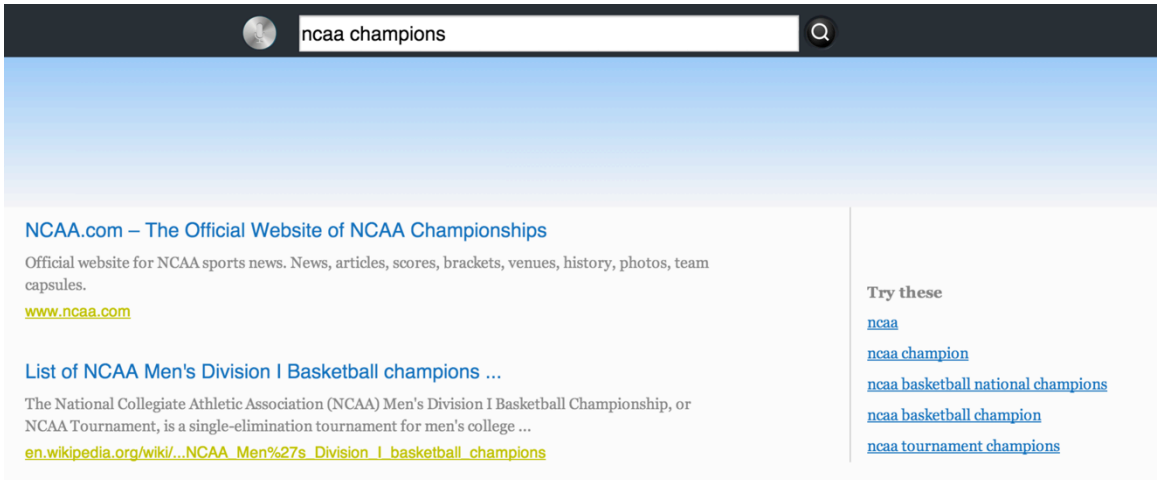


Figure 6.5

3. Try to narrow the results to basketball. Click on “NCAA basketball champion”.

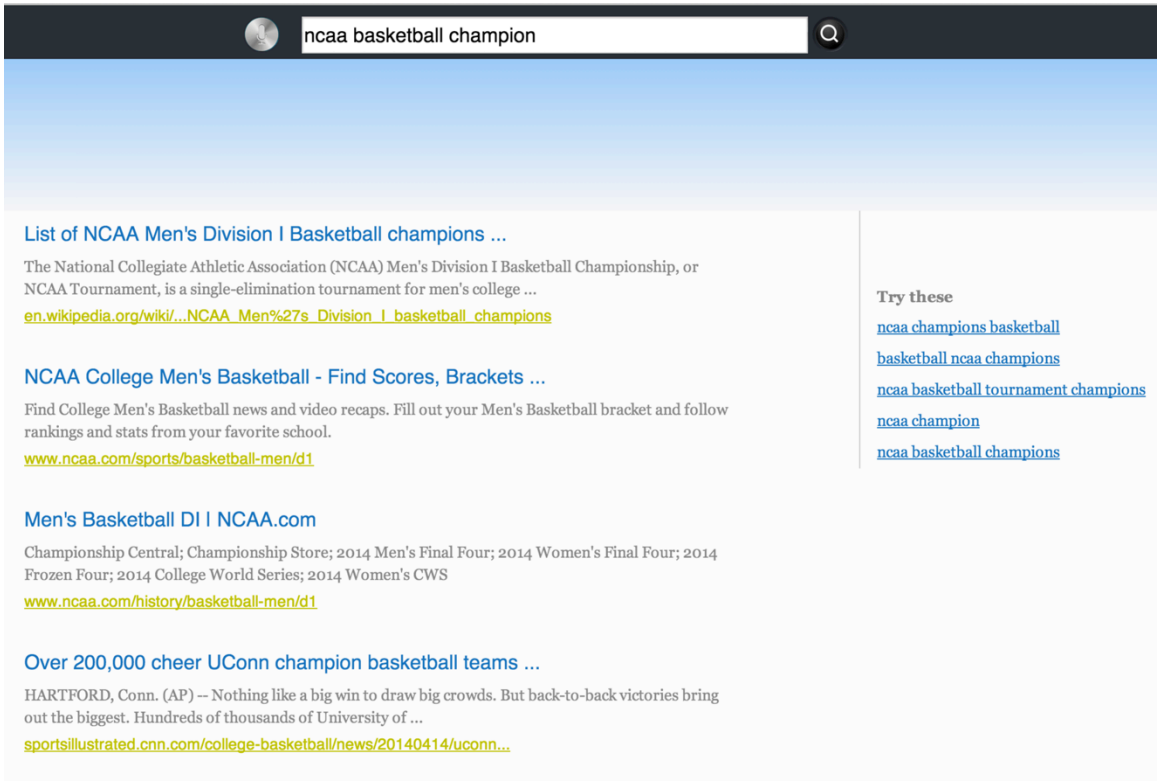


Figure 6.6

7 Evaluation of the System

In order to evaluate the outcome of the system, we conducted user studies and the process details are shown below.

User Study

✓ **Subjects**

We have six subjects from school of information science. Three students are international students who are not native English speaker and have some accent in speaking English. The other three students are native speakers. Expected queries are pre-designed and given to each subject to perform the voice search in our system. Each experiment lasts about half an hour.

✓ **Test Cases**

Table 8.1 Evaluation Tasks

Query/Task
1. "rap and crime"
2. "wrap and feel"
3. Try to find more information of Central Michigan University with abbreviation "CMU"
4. "field son"
5. "crying sun"

✓ **Observation**

In order to better evaluate on our system, the observation is necessary when the user is doing the test. In our evaluation, we recorded the total count of **operations** and **time** the user has taken. In details, the operations include inputting voice, changing query through featured lists and changing query by typing. Since the goal of our system is to reduce the user's efforts to IE correct, the less operations the better result is. Moreover, the count of operations of changing query by typing is 5 while the other operations are counted as 1, because the system is intended to enable the user to avoid typing. So, the count of operations is listed in Table 8.2.

Table 8.2 The Count of Operations

Operation	Count
Voice Input	1
Change Query by List (Clicking)	1
Change Query by Typing	5

✓ **Questionnaires**

Based on [5] *User Interface Usability Evaluation with Web-Based Questionnaires,* we designed our own questionnaires to evaluate the system. The users will be asked to evaluate the system based on the questionnaires and to provide a score from 0 to 7. The questions are shown in Table 8.3 – Table 8.5.

Table 8.3 *PERCEIVED EASE OF USE* Questions

1-1. Learning to operate the system would be easy for me	**Score 0 -7**

1-2. My interaction with the system would be clear and understandable	**0: Unlikely**
1-3. I would find it easy to get the system to do what I want it to do	**7: Likely**
1-4. I would find the system to be flexible to interact with	

Table 8.4 *PERCEIVED USEFULNESS* Questions

2-1. I would find the system useful in my task	**Score 0 -7**
2-2. Using the system would enhance my effectiveness on the job	**0: Unlikely**
2-3. Using the system in my job would increase my productivity	**7: Likely**
2-4. Using the system would improve my job performance	

Table 8.5 Other Questions

Question	Score 0	Score 7
3-1. Reading characters on the screen	Hard	Easy
3-2. Organization of information	Confusing	Very clear
3-3. Position of messages on screen	Inconsistent	Consistent
3-4. Prompts for input	Confusing	Very clear
3-5. Error messages	Unhelpful	Helpful
3-6. System speed	Too slow	Fast enough
3-7. Correcting your mistakes	Difficult	Easy

- ✓ **Results**
  - ■ **Subjective**

Table 8.6 Subjective Evaluation Result

Question	Lowest Score	Highest Score	Average Score	
			International	Native
1-1	5	7	5.67	6.33
1-2	6	7	7	6.67
1-3	5	7	6	6
1-4	7	7	7	7
2-1	4	7	6	5.33
2-2	5	7	6.33	5.67
2-3	5	6	5.66	5
2-4	4	5	5	4.33
3-1.	7	7	7	7
3-2.	6	7	6.33	7
3-3.	6	7	6.67	6.33
3-4.	5	7	6.33	6
3-5.	6	7	6.33	6.66
3-6.	4	6	5	5.33
3-7.	5	7	6	6.33

- **Objective**

Table 8.7 Observation (Objective) Result

Task	Average Count of Operations		Average Time Spend	
	International	Native	International	Native
1	4	3	7 min	4.5 min
2	2.33	2.67		
3	2	2		
4	3.33	2		
5	2.33	1.67		

✓ **Analysis**

Based on the subjective result, we can conclude that the users are not satisfied with the responding time. And the international speakers give higher scores than native speakers, which means our system provides better supports to the user who has accent in speaking English.

The observed results are interesting and show lots of information during the process. During the experiment, none of the user has typed to change the query. As is shown in the Table 8.7, the native users finished the tasks more quickly, because the native speakers have less recognition errors. According to the average count of operations, the counts for task 1, 2,4, 5 of native users are significantly smaller than the non-native speakers while the results are exactly same between non-native and native speakers for task 3. The speakers with accent need more operations to correct their errors. For example, the non-native speakers are always not able to clearly pronounce “son”, “sun”, and “song”, so they need more operations for task 5.

8 Conclusion

8.1 Summary

On the whole, we have found our system to be successful at assisting users with voice search. We were able to implement a system that dealt with the problems listed in the literature. [1][2] Our system first tried to help users by showing their most likely recognized words. It then shows words that sound similar. The system also lists related words and synonyms to help a user who may not be as familiar with a topic to reformulate a better search. We implemented several tools including voice to text in Chrome, Bing search and synonyms, and relative words from veryrelate.com to bring together a coherent,

easy-to-use system that allowed users to quickly search for what they were looking for in a familiar and user-friendly environment. The system we implemented guided users who were unfamiliar with a topic or found a topic too complex towards more accurate results. We certainly have improvements that can be made in the future but our system is a good first step towards making voice search work for the users.

8.2 Future Improvement

8.2.1 Mobile Compatibility

We can imagine implementing a mobile version in future. Nowadays mobile websites have become more popular. People want to use mobile search because of its portability. However, the interface design between mobile version and traditional website version could be large different.

One special character of mobile version is its webpage size. Mobile has a small screen so that it is inappropriate and impossible to display all of the reformulation units. In this case, we can come up with a solution by using tags. All the reformulations can be imbedded into tags and users click tags to show them when necessary. Also, we can keep using moving search bar with a header. This can still make the mobile version concise and aim to the result display.

8.2.2 Improvement of Interface Design

Although we try our best to make the interface concise, convenient and useful, it still has many ways to improve.

Firstly, the size of the block using to contain recognized alternatives should be varied with the number of alternatives. Sometimes the number of alternatives will be quite few or even none. Fix the size of that block could occupy a lot of space, which should better allocate to result items.

Secondly, we limit the number of items less than 5 or 6 in each group of formulation units. It just the offset between concise and utility decided based on our mind. In future, we should do more research on this part. Such as how many relatives should we display for the users. Moreover, the whole webpage seems a little bit monotonous, because it lack of the type of results. It only contains text. In future, we should add more type of results such as photos to make the content more abundant.

8.2.3 Improvement of Features

a) Homonyms Dictionary

Obviously, the current data for homonyms within the system is very limited. We just use some typical words for the demonstration of the idea. In the future, the dictionary of homonyms needs lots of supplements and improvement. For example, we can integrate an API to get a list of homonyms for each word.

There is an interesting website (https://words.bighugelabs.com/) which provides the feature to get a list of "sounds like" words. And it is still working on the implementation of such API. The API is not completed currently.

There are also some other useful resources (Table 9.1) that can provide much information and data for constructing the dictionary for homonyms.

Table 9.1 Resources for Homonyms

Resources	Description	Link
Big Huge Thesaurus	Provides various words related to the base word, such as sounds like, same rhymes, verbs and so on.	Link
Homophone List	An English Homophone Dictionary from Peter Suber, Philosophy Department, Earlham College	Link
CMU Pronouncing Dictionary	Machine-readable pronunciation dictionary for North American English	Link
Wordnik	Shows definitions for the word from multiple sources	Link

b) Functionalities

Because of the limited time, the functionalities of our system are not such perfect. Some expected improvements are shown as below.

✓ **Homonyms**

By now, the homonyms list enables the user to change the base word in query with the similar word. However, if there is more than one base word in the query, all of the base words will be changed into the chosen similar word. It is kind of awkward result. So, in the future, the list should be able to let the user to choose which word in the query to be changed by which similar word.

Another insufficiency in Homonym List is that the search_frequency in the data are static and do not change with the search actions. In the future, we would like to have a research on the popularity of each term in searching so that the ranking of homonyms can be based on the data that is more accurate and meaningful. Hence, in the future, we can try to look for a good API that can provide the search frequency of the term in popular search engines, or we can also design an algorithm to calculate the search frequency with the users' behaviors.

✓ **Relative Words**

The current system provides the relative words to the query and the user can choose to change the query by the relative word or append the relative word to the tail of the query. To be more efficient and effective, it should enable the user to append the relative words to either the head or the tail of the query.

Base Word	Relative Word	Current Result	Expected Result
Potato	Cheese	1. Potato Cheese	1. Potato Cheese
		2. Cheese	2. Cheese Potato
			3. Cheese

c) Efficiency

Based on the evaluation from the users, the response time of the system is a bit slow due to the waiting time for response from third-party APIs. In the future, we will try to find other strategies to substitute the API or find other APIs that are able to provide better response time. And also the process of API responses will need to be evaluated and improved to decrease the total processing time.

9 Reference

[1] Jiang, J., Jeng, W., & He, D. (2013). How Do Users Respond to Voice Input Errors? Lexical and Phonetic Query Reformulation in Voice Search. In Proceedings of the SIGIR'13.

[2] Jeng, W., Jiang, J., & He, D. Users' Perceived Difficulties and Corresponding Reformulation Strategies in Voice Search.

[3] https://words.bighugelabs.com/

[4] http://www.veryrelated.com/related.php

[5] http://hcibib.org/perlman/question.html

[6]https://onedrive.live.com/view.aspx?resid=9C9479871FBFA822!112&app=Word&wdo=2&authkey=!ANNnJQREB0kDC04

[7] http://www.zyvra.org/lafarr/hom.htm

[8] http://datamarket.azure.com/dataset/bing/synonyms