CSCI-GA 2580 Final Project
Query Suggestions and Instant Search Based on Wikipedia Corpus

Kentaro Hanaki   Yu-Chih Yu   Shuang Zhou

December 15, 2014

1   Introduction

As search engines are playing an increasingly important role, the expectations of search engines also increase. The historic purpose of search engines is finding relevant documents on the Internet, but nowadays this is far from enough.

Many advanced features are being incorporated into search engines, among which there are two interesting ones: query suggestion and instant search. Query suggestion is useful in two ways: First, users do not have to input the entire query and can choose from suggestions. Second, suggestions can provide a general idea about what users want to search when they are not sure.

Instant search perfectly complements query suggestion. With instant search, users can not only see the suggested "query", but also searching results, which allow users to be informed of what they are looking for, and thus allow them to refine their query based on retrieved results. Using this two techniques, we provide a more interactive search engine. And with rapid feedback, users can organize their queries more accurately and get more precise results.

This project implements a search engine that adopt this new interaction mode. In the indexing time, our search engine mines the Wikipedia corpus provided in the course assignments and store words and frequencies into Trie and the frequency of n-grams into suffix tree. In the serve time, it also accumulates the users' queries into Trie. For any partial query (prefix or words), our search engine finds most likely completion from these data structures and then recommends a list of relevant queries to users' input. Meanwhile, the results for the most probable query from the recommendation list along with document snippets are served instantly. All interactions are Ajax-based and no static elements are updated for queries, therefore, the traffic is minimized.

Our core data structure for query suggestion is Trie, which is essentially a prefix tree [1]. And it's a great data structure to store dynamic sets [?]. Query suggestion signals come from two sources: corpus statistics [2] and user log.

Instant search is first used in Google, and now adopted by some other popular search engines (e.g. Baidu). A lot of our instant search design is inspired by Google.

2 Design and Architecture

3 Implementation Details

4 Evaluation

In this section, we evaluate our query suggestion algorithms. The goal of our project is to approximate the query suggestions based on query logs using information from corpus. Therefore, we compare the results of our algorithm to the suggestions from the commercial web search engine. More specifically, we compare our result to the results from Google.

We examined the quality of completion in the following cases:

- The first case is when the first three characters of words are typed. Our search engine suggest single words in this case.

- The second case is when one or two words are typed. Our search engine suggests phrases based on the words entered.

In both cases, we compared the results with Google suggestions. More specifically, we took top 4 suggestions from both our search engine and Google and see if those suggestions agree. In order to avoid unnecessary personalization, we conducted the experiment after logging out from Google account. Also, our search engine can suggest previously searched queries, but we turned off this functionality.

Prefix	Our system suggestions	Google suggestions	Accuracy
alg-	algebraic algorithms **algorithm** **algeria**	algebra algebra calculator **algorithm** **algeria**	0.5
pra-	practice practices practical praised	prada prankdial pratt institute practice fusion	0

Figure 1: Results of suggestions from prefix. Bold red suggestions are the ones that show up both in our search engine and in Google.

Figure 1 shows the suggestions for two example prefixes ('alg-' and 'pra-'). 'alg-' achieved the accuracy of 0.5, and different suggestions are also similar ('algebraic' and 'algebra'). On the other hand, our system suggestions for 'pra-' are too general and differ a lot from Google's results. This stems from the fact that our suggestions are based purely on frequencies in Wikipedia corpus and not on query logs. Thus it can suggest words that appears frequently in English texts but are not searched by users.

Figure 2 shows the results for phrase suggestions. First of all, most of the suggested phrases in the experiment make sense, although there are some phrases that do not make sense ('michigan minnesota mississippi', which came from the list of states). Among the words we experimented, 'political' achieved 0.5 accuracy ('political parties' and 'political science' agreed). Accuracy for 'new' was 0.25, and disagreement

Word	Our system suggestions	Google suggestions	Accuracy
political	**political parties**	political cartoons	0.5
	political science	**political science**	
	political hip hop	political compass	
	political economy	**political parties**	
new	new york	**new york times**	0.25
	new york times	new york lottery	
	new york city	new york post	
	new zealand	new jersey transit	
michigan	michigan press	michigan	0
	michigan minnesota mississippi	michigan football	
	michigan state university	michigan state football	
	michigan state	michigan football schedule	

Figure 2: Results of phrase suggestions. Bold red suggestions are the ones that show up both in our search engine and in Google.

stems from the fact that Google uses location information (we searched from NYU and got 'new jersey transit').

Finally, there is an interesting difference in the suggestions for 'michigan'. Our search engine suggested press and university, although Google suggested phrases related to football. This is because Google uses query logs to improve the suggestions. Most of American people are interested college football for Michigan rather than press or university itself, and this interest is reflected on the suggestions by Google.

5 Contributions

- Kentaro Hanaki: He was responsible for ranking and evaluating suggestions. He implemented the ranking procedure and designed/conducted the evaluation for suggestions. He also implemented logging of user queries.

- Calvin Yu: He was responsible for data structures for query suggestions. He designed and implemented efficient data structures for building and scoring suggestions from query prefix/words.

- Shuang Zhou: He was primarily responsible for front end. More specifically, he designed graphic user interface, designed and implemented asynchronous request to the server using Ajax, and implemented snippets.

References

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. *Introduction to Algorithms, third edition*, June 2007.

[2] H. Bast and I. Weber. Type less, find more: fast autocompletion search with a succinct index. In *SIGIR*, pages 364-371, 2006.