Adapting to the User's Internet Search Strategy on Small Devices

Jean-David Ruvini
Bouygues e-lab
1 avenue Eugène Freyssinet
78061 St Quentin en Yvelines
jdruvini@bouygues.com

ABSTRACT

World Wide Web search engines typically return thousands of results to the users. To avoid users browsing through the whole list of results, search engines use ranking algorithms to order the list according to predefined criteria. In this paper, we present Toogle, a front-end to the Google search engine for mobile phones offering web browsing. For a given search query, Toogle first ranks results using Google's algorithm and, as the user browses through the result list, uses machine learning techniques to infer a model of her search goal and to adapt accordingly the order in which yet-unseen results are presented. We report preliminary experimental results that show the effectiveness of this approach.

Categories and Subject Descriptors

I.3.6 [**Computer Graphics**]: Methodology and Techniques – *Interaction techniques,* H.3.4 [**Information Storage and Retrieval**]: Systems and Software – *User profiles and alert services.*

General Terms

Algorithms, Experimentation, Human Factors.

Keywords

Internet search engine, mobile computing, adaptive interface, machine learning.

1. INTRODUCTION

Today Word Wide Web search engines like Google, among others, search documents for user specified keywords and return a list of document snippets (called results) where the keywords were found. With the vast amount of on-line information sources available on the web, this list typically contains thousands of results.

Because browsing through this list of results is often a tedious task, particularly on small devices offering web browsing (like mobile phones) which have limited display capabilities and to help users quickly locate relevant documents, search engines use ranking algorithms to order the list according to certain criteria (like number of keywords matched, proximity and frequency of keywords, etc.), presenting relevant documents in the top of the list and less relevant documents below. Alternatively, [2] proposed to learn the ranking function from clickthrough data (the logfile of results the users clicked on) of a group of users.

These ranking algorithms have two major limitations. First, they are global to all users and do not adapt to a specific user and to a specific search session or query. Second, they are exclusive in that they cannot be combined with or benefit from other ranking algorithms.

In this paper, we present Toogle, an intelligent front-end to the Google search engine that overcomes these limitations. Toogle first ranks results using Google algorithm and, as the user steps through the result list, uses machine learning techniques to infer a model of the user's search goal from clickthrough data and to adapt accordingly the order in which results are presented. Toogle is available for both desktop and small wireless browsers but has been mainly designed to improve usability of small devices.

2. TOOGLE : AN INTELLIGENT FRONT-END TO GOOGLE

As we mentioned above, Toogle, as Google, is accessible from desktop internet browsers and mobile phone offering web browsing, namely i-mode mobile phone. First introduced in Japan in February 1999 by NTT DoCoMo, i-mode is one of the world's most successful services offering wireless World Wide Web (WWW) access.

For both their desktop and i-mode versions there is almost no difference, from the user perspective, between Toogle and the Google search egine. Toogle is accessed from a web browser and its homepage is a clone of Google homepage. To query Toogle, the user just has to type in some keywords and to press the "Google Search" button exactly as he uses to do with Google. Similarly, Toogle results are displayed in the same format as Google results, grouped into pages that can be accessed individually from links located at the bottom of the current result page.

Figure 1(a)[1] shows the homepage of Toogle i-mode. Figure 1(b) and 1(c) show the result page of Toogle i-mode for the query "korean restaurant miami". Figure 1(b) presents the first lines of that result page; on Figure 1(c) the user has scrolled down this

IUI'03, January 12–15, 2003, Miami, Florida, USA.
ACM 1-58113-586-6/03/0001.

[1] The i-mode emulator shown in Figure 1 is I-MIMIC © X-9 DESIGN LAB (see http://www.x-9.com/).

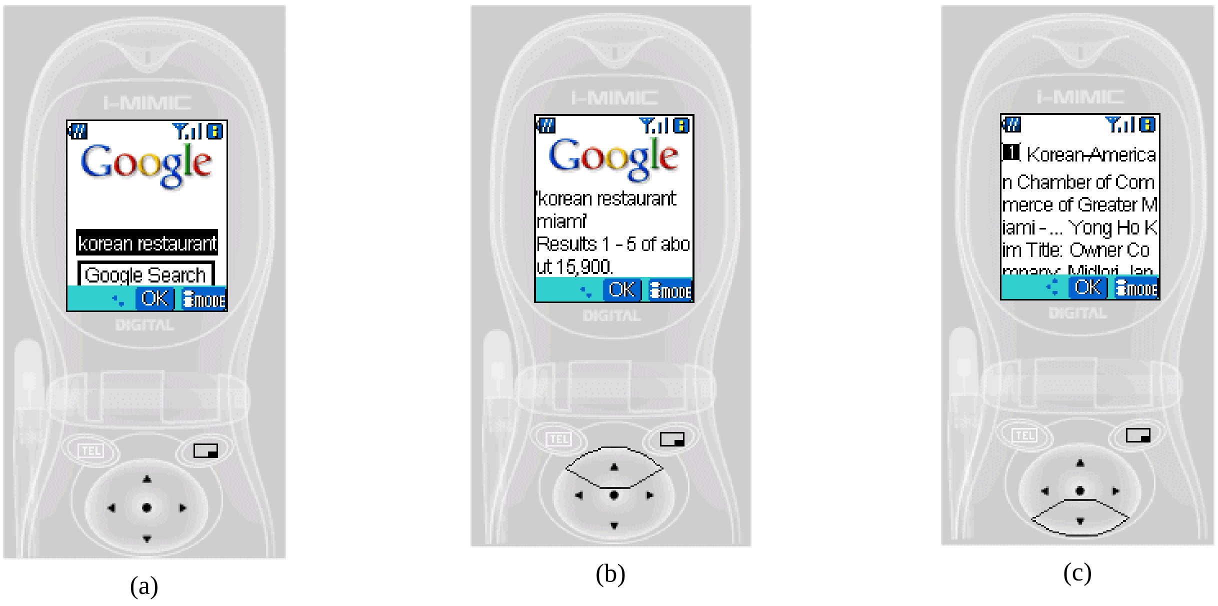
korean restaurant
Google Search
OK
'korean restaurant miami'
Results 1 - 5 of about 15,900.
1 Korean-American Chamber of Commerce of Greater Miami - ... Yong Ho Kim Title: Owner Company: Midlori Jap
(a)
(b)
(c)

Figure 1 The homepage 1(a) and a result page 1(b) and 1(c) of the i-mode versions of Toogle and Google.

page to review the first result proposed. The main difference between desktop and i-mode versions of Toogle (respectively Google) is the number of results displayed on each result page. It is 10 for the desktop browser version and 5 for the i-mode version.

Technically, Toogle acts as a proxy between Google and the user, using a web service programming toolkit released by the Google corporation.

3. ADAPTING TO THE USER'S SEARCH STRATEGY

Toogle is based on the observation that there is not an exact mapping between search queries and search goals. Since result lists typically present a lot of different information, users may type the same query for accessing different documents and, as a consequence, exhibit different browsing behaviors for that query. In other words, the browsing behavior of the user for a given query depends on his search goal (or interest). Toogle elaborates on this observation to infer, for a each search query, a model of the user interest and to adapt the list of results accordingly by reordering the list of results the user has not yet considered in order to present most relevant results first.

3.1 Building a model of the user's interest

Toogle goal is to build, for a given user U, a search query Q and its corresponding list of results L, the following target function:

$$ResultInterest_{U,Q,L} : Result \text{ ® } \{0,1\}$$

Given a result, the value of *ResultInterest*$_{U,Q,L}$ is interpreted as a measure of the interest the user has in it. A value of 1 indicates that he has a strong interest in it and is likely to click it to examine the corresponding document, whereas a value of 0 indicates that he is not likely to examine the document.

Toogle uses a machine learning approach to build the *ResultInterest*$_{U,Q,L}$ function. More precisely, because results are textual data (document snippets), it employs a text classifier to learn it. This approach requires to identify positive examples of results the user is likely to click and negative examples i.e. results he is not likely to click. Remind that Toogle (as Google) groups results into pages. For a given search query, Toogle considers, within all the result pages the user has visited, results he has clicked as positive examples and the results he has not clicked as negative examples. Suppose for example that, using an i-mode browser (5 results per page), the user has clicked the results ranked in position 3 and 5 in the list. Toogle considers results 3 and 5 as positive examples and results 1, 2, 4 as negative examples. Of course, result pages the user has not visited are not taken into account in the learning process (no example is extracted from them).

Once it has identified the examples, Toogle represents them in a text classifier exploitable form, namely the bag-of-words representation and invokes the text classifier to build the model of the user's interest. In its current implementation, Toogle learning algorithm is the Support Vector Machine [4]. Toogle builds a model or revises a learned model, whenever it identifies a new positive example that is to say whenever the user clicks a result.

3.2 Reordering the result list

To avoid disturbing the user by modifying the ranking of results on a page she has visited, Toogle reorders only pages she has not visited yet. This hidden reordering occurs whenever she requests a new result page and involves three steps.

First, Toogle queries Google for more results and loads them in memory. Since these results are obtained from Google, they are ordered according to the Google ranking criteria. Second, it uses the text classifier and the learned user's model to classify the loaded results. It then reorders the result list according to the predicted label : every result classified as a positive example is considered as a potentially interesting one and is moved to the top of the list. Toogle locally preserves Google ranking in that it does not perturb the relative ordering of positive examples and the relative ordering of negatives examples : if results *r* is ranked higher than *r'* in Google ordering and *r* and *r'* have the same predicted label then *r* is ranked higher than *r'* in Toogle ordering.

Table 1. Predictive accuracy and browsing gain of the i-mode version of Toogle for eight queries.

		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Mean
Positive examples		5	4	7	5	5	11	3	4	5.50
Negative examples		10	16	16	10	14	7	9	8	11.25
Predictive accuracy	NBC	22	93	77	50	50	58	71	14	54.48
	SVM	100	80	100	80	72	54	72	100	82.85
Browsing gain	Page	2	0	3	2	3	0	0	2	1.50
	Ratio	0	100	0	25	47	71	100	0	42.87

Third, Toogle displays the results to the user, presenting the most relevant at the top of the result page and the least relevant below.

As en example, suppose that, using an i-mode web browser (5 results per page), the user has requested result page number 2 (results 6 to 10) and that Toogle has predicted that results 7, 9 and 12 (according to the Google ordering) may be of interest to her. Toogle then displays a result page presenting (from top to bottom) results 7, 9, 12, 6 and 8. Results 6 and 8 are negative examples but since they are the negative examples with the highest rank in the Google ordering they are presented before any other negative example.

Of course, whenever the user submits a new search query Toogle forgets everything it has previously learned and start learning a new model specific to the new query and the new search goal.

4. EXPERIMENT

The goal of the experiment presented here is to show that, on i-mode, Toogle facilitates browsing through a list of search engine results by reducing user's efforts and information overload. More precisely, it is to answer the two following questions:

1. Can text classification algorithms successfully learn the user's interest from clickthrough data?

2. Does Toogle reduce the number of results users have to consider and the number of result pages they have to browse?

We conducted a preliminary experiment by submitting to two users (a researcher in our laboratory and a computer novice) eight search tasks using Google on the I-MIMIC i-mode emulator. To avoid query formulation bias the corresponding search queries were also given to the users. Using a proxy architecture we recorded users' browsing actions. Using these records we then simulated the users to feed Toogle with their browsing actions and analyzed Toogle behavior. For the eight tasks, the users exhibited the same browsing behavior (they clicked the same results).

The results of this experiment are presented in Table 1. Column labels Q1 to Q8 refer to the eight queries used in the experiment. The table comprises three parts. The first part, composed of the first two rows, lists the number of positive and negative examples of each query. The second part gives the predictive accuracy of Toogle for two learning methods: the Support Vector Machine (SVM) and the Naïve Bayes Classifier (NBC). We used the SVM-Light [1] implementation for the SVM, and the Bow toolkit [3] for NBC. The predictive accuracy is defined as the fraction of examples which label ("positive" or "negative") was correctly predicted by Toogle when trained on the examples of the first result page. The third part proposes an evaluation of the browsing gain for the end user of Toogle, in term of "page gain" and of gain "ratio" on the basis of the SVM algorithm. The page gain is defined as the difference between Google ordering and Toogle ordering in term of the number of result pages the user has to visit to be able to click the same results. The gain ratio measures how far from the optimal Toogle reordering is in term of ranks of the positive examples.

This Table shows that Support Vector Machines achieve good predictive accuracy (around 80%). This is an interesting finding because it suggests that SVM are able to learn a user's model even when few training data are available which is often the case in the area of User Modeling and Adaptive Interfaces. It also shows that there is a real gain in using Toogle (page gain is at least 2) for more than half of the queries (57%) and that Toogle reordering is, on the average, at 42% from the optimal.

5. SUMMARY

In this paper, we presented Toogle, a front-end to the Google search engine, that reduces search engine users information overload by adapting to the user's inferred search goal the order in which search results are presented. Toogle first ranks results using Google's algorithm and, as the user steps through the result list, uses machine learning techniques to build a model of his search goal and reorders the list of results accordingly. A preliminary experiment shows that Toogle performs well in practice on wireless phones offering WWW browsing, reducing substantially the number of result pages the user has to visit. Since it elaborates over an existing ordering and locally preserve this ordering, Toogle can benefit from any ranking algorithm.

6. REFERENCES

[1] Joachims T. SVM-Light Support Vector Machine; 1999. http://svmlight.joachims.org/.

[2] Joachims T. Optimizing Search Engine using Clickthrough Data. In Proceedings of the *ACM Conference on Knowledge Discovery and Data Mining*, ACM, 2002.

[3] McCallum A. K. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering, 1996. http://www.cs.cmu.edu/~mccallum/bow.

[4] Vapnik V. The Nature of Statistical Learning Theory. Springer-Verlag, New-York, 1995.